

Stołeczny Ośrodek
Elektronicznej
Techniki Obliczeniowej

INFORMATYKA mikrokomputerowa

Ireneusz Węgiel

OPROGRAMOWANIE PODSTAWOWE ZX SPECTRUM
DEVPAC 3 (GENS 3, MONS 3)

ZX-SPECTRUM

Warszawa 1986 r.

W książce omówiono zasady posługiwania się najbardziej popularnym obecnie zestawem programów przeznaczonych do pisania i uruchamiania programów o kodzie maszynowym na minikomputer
ZX SPECTRUM.
Współpracujące ze sobą asembler wraz z edytorem oraz monitor z programem uruchamiającym (ang. debugger) stanowią niezbędne narzędzie dla programistów piszących w kodzie wewnętrznym.
Znajomość zasad posługiwania się asemblerem może okazać się przydatna także dla czytelników pism mikrokomputerowych, którzy chcieliby uruchamiać na własnym komputerze zamieszczane na łamach tych pism programy.
Liczne przykłady i uwagi dotyczące sposobu posługiwania się programami pozwalają na szybkie opanowanie prezentowanego materiału.

Ireneusz Węgiel

Oprogramowanie podstawowe ZX SPECTRUM - DEVPAC 3 / GENS 3, MONS 3 /

WSTĘP

Pakiet programów znany pod nazwą "DEVPAC 3" angielskiej firmy Hisoft zawiera dwa programy:
"GENS3" - program tłumaczący zapis symboliczny rozkazów mikroprocesora Z 80 na kod wynikowy (tzw. asembler) oraz prosty i wygodny edytor przeznaczony do wpisywania tekstów własnych programów użytkownika, oraz
"MONS3" - program przeznaczony do uruchamiania (ang. debugger) i przeglądania programów w kodzie maszynowym (monitor).

W pierwszej części przedstawiono zasady działania asemblera oraz sposób posługiwania się edytorem. Opis dotyczy programu "GENS3M21" oraz poprzedniej, najbardziej popularnej w Polsce wersji "GENS3". Podobnie postąpiono w części drugiej, gdzie prezentowana jest jedna z najnowszych wersji monitora - "MONS3M21", a także jego poprzednik - "MONS3".

Większość informacji dotyczących nowszych wersji (M21) odnosi się także do ich poprzedników. Różnice między wersjami mające istotne znaczenie dla użytkowników programów GENS3 i MONS3, zostały zaznaczone w tekście instrukcji. Najważniejsze zmiany wprowadzone przez firmę Hisoft w wersjach "M21", to:

- dołączenie programu ładującego "G/M 3M21", który ładuje MONS3M21 od adresu 30000 a GENS3M21 od adresu 37000; "G/M 3M21" zmienia kolor tła i obrzeża na czarny, a kolor atramentu na biały, co sprawia, że korzystanie z programów jest mniej męczące
- sygnalizowanie przyciśnięcia klawisza dźwiękiem (keyboard click)
- likwidacja zmiennej wielkości bufora w programie GENS3M21

oraz szereg drobnych zmian ułatwiających posługiwanie się

programami, które zostały omówione w tekście instrukcji.

Uwaga:

znak "#" oznacza, że następująca po nim liczba jest zapisana w systemie szestnastkowym.

Część I

G E N S 3 M 2 1

ASEMBLER I EDYTOR

ROZDZIAŁ 1. URUCHOMIENIE PROGRAMU

Program GENS3M21 zajmuje około 10 KB pamięci (GENS3 - ok. 7 KB) włącznie z własnym, wewnętrznym stosem. Użytkownik może wczytać program w dowolne miejsce w pamięci. Należy jednak pamiętać aby zapewnić wystarczającą ilość pamięci na tekst programu użytkownika oraz tablicę symboli, które umieszczane są bezpośrednio po programie GENS3M21. Z tego względu wygodnie jest ładować program na początku pamięci.

1.1 Ładowanie i uruchamianie programu GENS3M21

GENS3M21 można załadować przy użyciu programu ładującego "G/M·3M21", który umieszcza program od adresu 37000 lub za pomocą komendy:

LOAD "GENS3M21" CODE aa

gdzie aa oznacza adres, od którego chcemy umieścić program. Uruchomienie programu odbywa się poprzez skok do adresu początkowego aa:

RANDOMIZE USR aa

Po uruchomieniu programu automatycznie zgłasza się edytor - patrz rozdział 3.
Ponowne uruchomienie programu GENS3M21 (tzn. po powrocie do systemu operacyjnego ZX SPECTRUM) może nastąpić w ten sam sposób, czyli:

RANDOMIZE USR aa

jest to tzw. ciepły start; nie powodujący zniszczenia poprzednio wpisanego tekstu programu użytkownika, albo:

RANDOMIZE USR aa+3

- zimny start, po którym tekst programu użytkownika zostanie zniszczony a argumentom komend edytora zostaną przypisane wartości początkowe.

Przykład:

załóżmy, że chcemy załadować program GENS3M21 od adresu 24000 aby zostawić ok. 30 KB wolnej pamięci na tekst własnego programu.

Ładowanie programu

LOAD "GENS3M21" CODE 24000

Uruchomienie programu i ciepły start

RANDOMIZE USR 24000

Zimny start

RANDOMIZE USR 24003

1.2 Ładowanie i uruchamianie programu GENS3

Program GENS3 ładujemy za pomocą komendy:

LOAD "GENS3" CODE aa

gdzie aa jest adresem, od którego chcemy umieścić program. Po załadowaniu GENS3, wejście do programu odbywa się poprzez skok do adresu początkowego aa:

RANDOMIZE USR aa

Program zgłasza się pytaniem o wielkość bufora:

"Buffer size?"

na które odpowiadamy wpisując cyfrę od Ø do 9 włącznie i ENTER lub samo ENTER (co odpowiada cyfrze 4). Cyfra, którą wpisujemy, określa wielkość bufora w jednostkach równych 256 bajtom; zero spowoduje utworzenie bufora o wielkości 64 bajtów. Bufor jest wykorzystywany przy nagrywaniu i wczytywaniu tekstów programów przeznaczonych do natychmiastowej asemblacji - szczegóły w rozdziale 2.9, komenda *F.

Wielkość bufora możemy określić tylko raz, po pierwszym uruchomieniu programu.

Ponowne uruchomienie programu GENS3 odbywa się w następujący sposób:

RANDOMIZE USR aa+56 - zimny start

albo

RANDOMIZE USR aa+61 - ciepły start

Uwaga: skok do adresu początkowego programu GENS3 może nastąpić tylko raz po uruchomieniu programu.

Przykład:

załóżmy, że chcemy załadować GENS3 od adresu 29944.

Ładowanie programu

```
LOAD "GENS3" CODE 29944
```

Uruchomienie programu

```
RANDOMIZE USR 29944
```

Zimny start

```
RANDOMIZE USR 30000
```

Ciepły start

```
RANDOMIZE USR 30005
```

ROZDZIAŁ 2. OPIS PROGRAMU GENS3M21

2.1 Zasada działania asemblera

GENS3M21 jest szybkim, dwuprzebiegowym asemblerem Z80. Dokonuje asemblacji wszystkich standardowych mnemoników rozkazów mikroprocesora Z80. Lista wszystkich mnemoników Z80 została zamieszczona w załączniku 2.

Po wywołaniu asemblera (za pomocą komendy "A" edytora - - patrz: rozdział 3.2) należy określić wielkość tablicy symboli (patrz: rozdział 2.5) w odpowiedzi na pytanie "Table size:". Wpisujemy liczbę dziesiętną i ENTER lub samo ENTER. W tym ostatnim przypadku asembler wybierze wielkość tablicy proporcjonalnie do długości tekstu programu użytkownika, co na ogół wystarcza.

Uwaga: używając komendy *F (patrz: rozdział 2.9) może zajść potrzeba określenia większej niż normalnie tablicy symboli - - asembler nie może przewidzieć długości tekstu wczytywanego z taśmy.

Następnie pojawia się pytanie "Options:", co umożliwia wybranie jednej lub kilku z następujących opcji:

1 wyświetla tablicę symboli po drugim przebiegu asemblera
2 asemblacja bez generowania kodu wynikowego
4 asemblacja bez listingu tekstu programu
8 listing asemblera skierowywany jest na drukarkę
16 umieszcza kod wynikowy, jeżeli jest generowany, po tablicy symboli. Licznik Adresów przyjmuje wartości podawane przez ORG, tak więc można umieścić bezpośrednio po tablicy symboli kod wynikowy przeznaczony do wykonywania w

w innym miejscu pamięci.

32 wyłącza sprawdzanie, w którym miejscu w pamięci jest umieszczony kod wynikowy - opcja stosowana w celu przyspieszenia asemblacji.

W odpowiedzi na "Options:" możemy wpisać:

ENTER - żadna z opcji nie zostanie wybrana

numer wybranej opcji

sumę numerów wybranych opcji

Po udzieleniu odpowiedzi na powyższe pytania, wyświetlany jest nagłówek asemblera:

```
*HISOFT GENS3M2 ASSEMBLER*
       ZX SPECTRUM

Copyright (C) HISOFT 1983,4
All right reserved
```

i zaczyna się asemblacja.

Asemblacja jest dwuprzebiegowa: podczas pierwszego przebiegu GENS3M21 sprawdza poprawność składniową tekstu i tworzy tablicę symboli, w drugim przebiegu generowany jest kod wynikowy (jeśli nie została wybrana opcja numer 2).

Podczas pierwszego przebiegu wyświetlane są jedynie wiersze zawierające błąd wraz z podanym numerem błędu (patrz: załącznik 1), np.:

```
120        SBV   HL,DE
*ERROR* 2                     błąd numer 2.
```

W przypadku wykrycia błędu, asemblacja zostaje zatrzymana. Po naciśnięciu "E" nastąpi powrót do edytora, a dowolnego innego klawisza - kontynuowanie asemblacji od następnego wiersza.

Po pierwszym przebiegu wyświetlany jest komunikat:

Pass 1 errors: n (pierwszy przebieg, n błędów)

Jeśli choć jeden błąd został wykryty, asembler nie przystąpi do generowania kodu wynikowego. Jeżeli w tekście programu są odwołania do niezadeklarowanych etykiet, ukazują się komunikaty, np.:

WARNING START absent

(UWAGA brak etykiety o nazwie START)

Podczas drugiego przebiegu, oprócz generowania kodu wynikowego, wyświetlany jest listing asemblera (jeśli nie został wyłączony przez opcję nr 4 lub komendę *L- asemblera).
Postać listingu asemblera jest następująca:

```
kolumny
ekranu   1  4 6        15    21   26   31

         D5AA CDB2D6     720 PODPR1 <------ etykieta
                             CALL #D6B2
```

Opisy do listingu (strzałki): D5AA – wartość Licznika Adresów; CDB2D6 – format rozkazu; 720 – numer wiersza; PODPR1 – etykieta; CALL – mnemonik; #D6B2 – argumenty.

Na początku każdej linii znajduje się wartość Licznika Adresów, chyba, że w danym wierszu znajduje się jeden z pseudo-mnemoników: ORG, EQU lub ENT (patrz: rozdział 2.7) - wtedy wyświetlana w tym miejscu liczba jest równa wartości znajdującej się w polu argumentów rozkazu. System, w jakim wyświetlane są te liczby możemy zmienić na dziesiętny umieszczając w tekście programu komendę *D+ asemblera (patrz: rozdział 2.9). Po adresie wyświetlana jest zawartość komórek pamięci (maksimum czterech - kod najdłuższego rozkazu mikroprocesora Z80) składająca się na jeden rozkaz kodu wynikowego. Fragment listingu reprezentujący format rozkazu możemy usunąć za po-

mocą komendy *C- asemblera (patrz: rozdział 2.9).
Następną liczbą jest numer wiersza - od 1 do 32767. W kolumnach 21-26 wyświetlane jest 6 pierwszych znaków etykiety. Pozostała część wiersza programu zajmuje kolejne linie ekranu, począwszy od 21 kolumny: mnemonik (kolumny 21-24), argumenty (od 26 kolumny) oraz komentarze.
Taki format listingu poprawia czytelność wydruku gdy w jednej linii można wyświetlić maksimum 32 znaki. Istnieje możliwość zmiany formatu listingu asemblera poprzez wpisanie do odpowiednich komórek pamięci, zajmowanej przez GENS3M21 liczb określających:

a. w której kolumnie kończy się pierwsza linia ekranu. Wpisujemy numer kolumny pomniejszony o 5. Początkowo w komórce pamięci o adresie podanym poniżej, znajduje się liczba 22, co oznacza, że pierwsza linia ekranu kończy się w kolumnie 27. Zmieniając tę liczbę na Ø uzyskamy pełną szerokość wydruku.

   Adres: początek programu GENS3M21+51

b. w której kolumnie zaczyna się każda kolejna linia ekranu.

   Adres: początek programu GENS3M21+52

c. ilość znaków wyświetlanych w drugiej i każdej następnej linii ekranu.

   Adres: początek programu GENS3M21+53

Na przykład, jeśli chcemy, aby pierwsza linia ekranu zawierała 14 znaków (tj. tylko adres i format rozkazu), a następne zaczynały się od pierwszej kolumny i wypełniały całą linię, należy wprowadzić następujące instrukcje z BASICa (załóżmy, że program GENS3M21 został załadowany od adresu

37000):

POKE 37051,9	14-5 = 9
POKE 37052,1	musi być co najmniej jedna spacja na początku każdej następnej linii
POKE 37053,31	31 znaków

Powyższe modyfikacje zostaną uwzględnione pod warunkiem, że nie będziemy stosować komendy *C- asemblera.

Listing asemblera można zatrzymywać wciskając SPACE (w programie MONS3 - BREAK), aby następnie wrócić do edytora - klawisz "E", lub kontynuować asemblację - dowolny klawisz.

Jedynymi błędami, jakie mogą wystąpić podczas drugiego przebiegu asemblera, są *ERROR* 10 i "Bad ORG!" (patrz: załącznik 1). "Bad ORG!" występuje, jeśli generowany kod wynikowy mógłby zniszczyć GENS3M21, tekst programu użytkownika lub tablicę symboli. Błąd ten przerywa asemblację i przekazuje sterowanie do edytora. Po *ERROR* 10 asemblacja może być kontynuowana. Po drugim przebiegu wyświetlany jest komunikat:

    Pass 2 errors: n   (drugi przebieg, n błędów)

Następnie pojawiają się ostrzeżenia o niezadeklarowanych etykietach oraz informacja o wielkości tablicy symboli w porównaniu z wielkością wstępnie przydzieloną:

    Table used: x from y

gdzie x oznacza wielkość tablicy symboli a y wielkość zarezerwowanego obszaru.

W końcu, jeśli została prawidłowo użyta dyrektywa ENT (patrz: rozdział2.7), wyświetlany jest komunikat:

    Executes: aa

gdzie aa oznacza adres, od którego zostanie wykomany kod wynikowy gdy użyjemy komendy edytora "R". (patrz: rozdział 3.2).

Gdy została wybrana opcja numer 1, asembler wyświetli listę etykiet wraz z odpowiadającymi im wartościami. Po zakończeniu asemblacji sterowanie jest przekazywane do edytora.

2.2 Postać wierszy programu

Każdy wiersz programu użytkownika przeznaczony do asemblacji przez GENS3M21 powinien mieć następujący format:

NUMER WIERSZA	ETYKIETA	MNEMONIK	ARGUMENTY	KOMENTARZ
100	START	LD	DE, (LISTA)	; inicjalizuj wskaźnik

Wiersze programu są analizowane przez asembler w następujący sposób:

Badany jest pierwszy znak w wierszu (po numerze wiersza i spacji) i następne kroki zależą od tego, jaki to jest znak. Jeśli pierwszym znakiem jest:

";"	cały wiersz jest traktowany jako komentarz
"*"	spodziewany jest symbol jednej z komend asemblera (patrz: rozdział 2.9). Pozostałe znaki są traktowane jako komentarz.
ENTER (CHR$ 13)	czyli znak końca wiersza - wiersz jest pomijany
" "	spacja lub znak tabulatora - pierwszy znak nie będący spacją ani znakiem tabulatora traktowany jest jako początek mnemonika Z80.
inny znak	asembler traktuje to jako początek etykiety

Jeśli etykieta jest prawidłowa lub jeśli pierwszym znakiem w wierszu jest spacja (znak tabulatora), to następny znak nie będący spacją (znakiem tabulatora) traktowany jest jako początek mnemonika Z80 (patrz: załącznik 2) o długości do 4 znaków i zakończony spacją (znakiem tabulatora) lub znakiem

końca wiersza. Jeśli mnemonik jest prawidłowy i wymaga argumentów, wtedy następujące po nim spacje są pomijane i przetwarzane jest pole argumentów.
Wiersze mogą zawierać same etykiety (w celu zwiększenia czytelności tekstu). Komentarze, poprzedzone znakiem ";", mogą występować w dowolnym miejscu po argumentach lub, jeśli mnemonik nie wymaga argumentów, po mnemoniku.

2.3 Etykiety

Podczas pisania programów w postaci symbolicznej bardzo często stosuje się tzw. adresy symboliczne lub stałe noszące ogólną nazwę etykiet. Etykiety reprezentują 16 bitów informacji. Stosowanie etykiet uwalnia programistę od kłopotów jakie stwarzałoby obliczanie adresów komórek pamięci, w jakich znajdują się poszczególne rozkazy. Poza tym etykiety znacznie zwiększają czytelność programów, zaś w przypadku konieczności wstawienia lub usunięcia rozkazów z programu, programista nie musi zmieniać adresów i skoków.
Wartości etykiet obliczane są w pierwszym przebiegu asemblera lub podawane przez programistę (patrz: dyrektywa EQU - rozdział 2.7). Gdy etykieta nie zostanie zadeklarowana, pojawi się ostrzeżenie (*WARNING*), a jako wartość etykiety jest przyjmowana bieżąca wartość Licznika Adresów. Jeśli etykieta reprezentuje wartość większą niż 8 bitów i zostanie użyta w miejscu, gdzie spodziewana jest stała 8-bitowa, wystąpi błąd: *ERROR* 10 (w drugim przebiegu asemblera), np.:

```
INDEKS EQU   #10AB
       LD    B,INDEKS
```

Etykiecie o nazwie INDEKS została przypisana wartość #10AB,

natomiast rozkaz LD B,n wymaga argumentu 8-bitowego. Jeśli w tego typu przypadkach asemblacja zostanie doprowadzona do końca,to zostanie przyjęta wartość mniej znaczącego bajtu etykiety (#AB).

Nazwa etykiety może zawierać dowolną liczbę znaków, lecz tylko pierwsze 6 znaków jest ważnych; te sześć znaków musi stanowić jedyną (niepowtarzalną) nazwę, ponieważ etykieta nie może być deklarowana ponownie (*ERROR* 4). Etykietą nie może być słowo zastrzeżone (patrz: załącznik 2), ale może ono być użyte jako część nazwy etykiety. Nazwa etykiety może składać się ze znaków: cyfry (od 0 do 9), wielkie i małe litery (od a do Z) oraz $, /, [,], ↑, _, £. Pierwszym znakiem musi być litera. Przykłady poprawnych etykiet:

```
KONIEC
koniec
A[1]
adr_10
RRCA            RRCA nie jest słowem zastrzeżonym
b/z
```

Alfabetyczny spis wszystkich etykiet użytych w programie możemy otrzymać wybierając opcję nr 1.

2.4 Licznik Adresów

W Liczniku Adresów przechowywany jest bieżący adres komórki pamięci, do której wpisuje się kod rozkazu. Licznik Adresów umożliwia przypisanie wartości (równej adresowi) poszczególnym etykietom i wprowadzenie ich do tablicy symboli. Wartości Licznika Adresów podawane są w listingu asemblera. Licznik Adresów może być ustawiony na dowolną wartość za pomocą dyrektywy ORG (patrz: rozdział 2.7). Jeżeli dyrekty-

wa ORG nie zostanie użyta, Licznik Adresów jest ustawiany na adres pierwszego bajtu po tablicy symboli. Wartość Licznika Adresów jest dostępna w programie poprzez symbol "$", np. : LD HL,$+12 spowoduje wygenerowanie kodu odpowiadającego bieżącej wartości Licznika Adresów +12 i taka wartość zostanie załadowana do rejestru HL podczas wykonania programu.

2.5 Tablica symboli

Etykieta napotkana po raz pierwszy podczas asemblacji jest umieszczana w tablicy symboli wraz z dwoma wskaźnikami określającymi, po zakończeniu asemblacji, jej miejsce w porządku alfabetycznym. Gdy dana etykieta znajduje się w polu etykiet (została zadeklarowana), jej wartość (równa bieżącej wartości Licznika Adresów lub wyrażeniu po dyrektywie EQU) jest umieszczana w tablicy symboli.
Tablica symboli jest zorganizowana w postaci drzewa binarnego, co umożliwia bardzo szybkie wprowadzanie i wyszukiwanie poszczególnych symboli. Jedna etykieta zajmuje, w zależności od długości jej nazwy, od 8 do 13 bajtów w tablicy symboli. W tablicy symboli umieszcza się tylko 6 pierwszych znaków nazwy etykiety, w celu ograniczenia miejsca zajmowanego przez etykietę. Jeśli etykieta jest deklarowana więcej niż jeden raz, wyświetlany jest komunikat błędu *ERROR* 4.

2.6 Wyrażenia

Asembler GENS3M21 akceptuje oprócz stałych liczbowych i etykiet jako argumentów, również wyrażenia. Jest to istotne ułatwienie dla programisty, gdyż nie musi on obliczać wartości

wyrażeń przed wpisywaniem programu. Wartości wyrażeń są obliczane podczas pierwszego przebiegu asemblera. Wyrażenie składa się z czynnika lub kilku czynników oddzielonych operatorami:

czynnik stała dziesiętna, np. 6128
stała szesnastkowa - poprzedzona znakiem #, np. #40AF
stała dwójkowa - poprzedzona znakiem %, np. % 1001011
stała znakowa - ujęta w cudzysłów, np. "\"
etykieta, np. Bufor1
$ - oznacza bieżącą wartość Licznika Adresów

GENS3M21 uwzględnia tylko 16 najmniej znaczących bitów liczby (tzn. MOD 65536) - np. #FC84E stanie się #C84E a 68749 - - 3219.

operator	+	dodawanie
	-	odejmowanie
	&	iloczyn logiczny - AND
	@	suma logiczna - OR
	!	suma modulo dwa - XOR
	*	mnożenie całkowite
	/	dzielenie całkowite - DIV
	?	modulo - MOD tzn. a?b = a-(a/b)*a

Wszystkie operatory posiadają równy priorytet - wyrażenia są obliczane od lewej do prawej strony.
Wyrażenie ujęte w nawiasy jest traktowane jako adres pamięci. Np. LD IX, (tabl+8) jest rozkazem przesłania do rejestru IX zawartości komórek pamięci o adresach tabl+8 i tabl+9.
Przykłady prawidłowych wyrażeń (zakładamy, że etykieta o nazwie STO = 100):

wyrażenia	wartość wyrażenia wyliczona przez asembler
#2AAA+STO	#2B0E
%110110 & %10001	%1000

```
"W" ! 12+STO             #BF
12+3-6/11                2
%111 @ "_"               #5F
($+STO ? 3000)
```

UWAGA: spacje można umieszczać między czynnikiem a operatorem ale nie wewnątrz czynników.

Jeśli wartość mnożenia będzie większa niż 32767, to wystąpi błąd *ERROR*15, a dzielenie przez zero spowoduje błąd *ERROR*14 - w innych przypadkach nadmiar będzie ignorowany.

W działaniach arytmetycznych stosowany jest kod U2, tzn. liczby większe od 32767 (#7FFF) są traktowane jako ujemne, np. 52000 = -13536 (52000-65536).

Argumentami rozkazów skoku względnego (JR i DJNZ) mogą być adres skoku (liczba lub etykieta) lub wyrażenie "$± przesunięcie względne". W pierwszym przypadku asembler wyliczy odpowiednie przesunięcie i wstawi je do kodu. Przesunięcie obliczane jest jako różnica podanej wartości argumentu i wartości Licznika Adresów dla następnego rozkazu (czyli $+2), np.:

```
B3CE  180E          90        JR    #B3DE
```

Przesunięcie (#0E) zostało obliczone w następujący sposób:

```
#B3DE-(#B3CE+2)=#0E
```

W drugim przypadku adres skoku względnego będzie obliczany względem wartości Licznika Adresów bieżącego rozkazu. Przesunięcie względne może więc zawierać się w granicach od -126 do +129 włącznie. Aby nie wprowadzać zamieszania, rozkaz skoku względnego zapisuje się zwykle jako

```
JR   $+2+d
```

gdzie d oznacza przesunięcie względne, które zawiera się w granicach od -128 do +127 włącznie.

Jeśli podane lub wyliczone przesunięcie względne będzie poza dozwolonym zakresem, to wyświetlony zostanie komunikat błędu: *ERROR* 10.

2.7 Dyrektywy asemblera

Asembler GENS3M21 rozpoznaje, oprócz mnemoników rozkazów mikroprocesora Z80, pseudo-mnemoniki, czyli tzw. dyrektywy asemblera. Dyrektywy nie mają wpływu na mikroprocesor podczas wykonywania programu tzn. nie są umieszczane w kodzie wynikowym a służą jedynie do przekazywania pewnych informacji asemblerowi. Działania podjęte przez asembler w wyniku tych informacji zmieniają w pewien sposób kod wynikowy generowany przez GENS3M21. Pseudo-mnemoniki są traktowane podobnie jak mnemoniki rozkazów; mogą być poprzedzone etykietą, a po nich może występować komentarz. Dyrektywy są następujące:

EQU wyrażenie

musi być poprzedzone etykietą. Przypisuje podanej etykiecie wartość wyrażenia. Wyrażenie nie może zawierać symbolu, któremu nie została jeszcze przypisana żadna wartość. (*ERROR*13)

ORG wyrażenie

ustawia Licznik Adresów na wartość wyrażenia; oznacza to, że kod wynikowy zostanie umieszczony od adresu podanego przez ORG. Jeśli nie zostały wybrane opcje 2 lub 16 a wartość wyrażenia jest taka, że generowany kod wynikowy mógłby zniszczyć program GENS3M21, tekst programu użytkownika lub tablicę symboli, to wyświetlany jest komunikat "Bad ORG!" i asemblacja zostaje przerwana. W takim wypadku możemy dokonać asemblacji ponownie wybierając opcję nr 16. Kod wynikowy zostanie umieszczony po tablicy symboli ale Licznik Adresów

przyjmie wartość podaną przez ORG. Adres początku obszaru pamięci, w którym umieszczony został kod wynikowy możemy obliczyć dodając następujące wielkości: adres końca tekstu (patrz: komenda X edytora, rozdział 3.2) oraz wielkość tablicy symboli + 2. Wygenerowany w ten sposób kod wynikowy należy przed wykonaniem przesunąć do miejsca określonego przez ORG.

ENT wyrażenie

ustala wartość adresu, od którego zostanie wykonany kod wynikowy po komendzie "R" edytora (patrz: rozdział 3.2), np.:

ENT 42183	- wykonanie kodu od adresu 42183
ENT $	- wykonanie kodu od adresu w Liczniku Adresów

Uwaga: jeżeli została wybrana opcja 16, dyrektywa ENT nie będzie wykonana.

DEFB wyrażenie, wyrażenie,...

wartość danego wyrażenia może być co najwyżej 8-bitowa. Do komórki pamięci adresowanej przez Licznik Adresów zostanie wpisana wartość danego wyrażenia, po czym Licznik Adresów zostanie zwiększony o 1. Te operacje są powtarzane dla każdego wyrażenia.

DEFW wyrażenie, wyrażenie,...

słowo (2 bajty) równe wartości wyrażenia zostanie wpisane do komórek pamięci adresowanych przez Licznik Adresów (mniej znaczący bajt najpierw), po czym Licznik Adresów zostanie zwiększony o 2. Te operacje są powtarzane dla każdego wyrażenia.

DEFS wyrażenie

zwiększa Licznik Adresów o wartość wyrażenia - równoznaczne

zarezerwowaniu obszaru pamięci o wielkości równej wartości wyrażenia.

DEFM "c"

"c" jest ciągiem znaków o długości n; do n komórek pamięci począwszy od adresu w Liczniku Adresów zostaną wpisane poszczególne znaki ciągu "c" w kodzie ASCII. Długość ciągu może teoretycznie wynosić 255 znaków, ale w praktyce limitowana jest przez długość wiersza, który możemy wprowadzić za pomocą edytora. Pierwszy znak ciągu (") nie jest wpisywany podobnie jak ostatni, którym może być cudzysłów (") lub znak końca wiersza (ENTER).

2.8 Warunkowe pseudo-mnemoniki

Warunkowe pseudo-mnemoniki pozwalają na wyłączenie części tekstu programu z procesu asemblacji:

IF wyrażenie

jeżeli wartość wyrażenia jest równa zero, asemblacja zostaje wyłączona (następne wiersze programu są pomijane przez asembler) aż do napotkania ELSE lub END. Jeśli wartość wyrażenia jest różna od zera, asemblacja jest komtynuowana normalnie.

ELSE

włącza i wyłącza asemblację. Jeśli przed napotkaniem ELSE asemblacja dokonywała się normalnie (była włączona), to zostanie wyłączona i odwrotnie.

END

powoduje włączenie asemblacji.

Uwaga: warunkowe pseudo-mnemoniki nie mogą być zagnieżdżane; zachowanie asemblera po napotkaniu zagnieżdżonych IF jest nieokreślone.

2.9 Komendy asemblera

Komendy asemblera, podobnie jak dyrektywy, nie mają wpływu na mikroprocesor podczas wykonywania programu, ponieważ nie są umieszczane w kodzie wynikowym. Co więcej, w przeciwieństwie do dyrektyw asemblera, nie wpływają również na generowanie kodu wynikowego (oprócz komendy *F), - służą jedynie do modyfikacji listingu asemblera.

Komenda asemblera składa się z gwiazdki "*" (która musi być pierwszym znakiem w wierszu) oraz wielkiej litery, a komendy "L", "C" i "D", dodatkowo ze znaku "+" lub "-".

W wierszu można umieścić tylko jedną komendę; pozostałe znaki traktowane są jako komentarz.

Komendy asemblera są następujące:

*L- wyłącza listing od danego wiersza

*L+ listing zostaje włączony począwszy od tego wiersza

*C- skraca listing asemblera począwszy od następnego wiersza. Skrócony listing nie zawiera przedstawienia kodu wynikowego. Zostaje w ten sposób zaoszczędzonych 9 znaków, co umożliwia zmieszczenie prawie każdego wiersza programu w jednej linii ekranu poprawiając tym samym czytelność listingu.

*C+ przywraca pełną postać listingu opisaną w rozdziale 2.1

*D+ powoduje wyświetlenie wartości Licznika Adresów w systemie dziesiętnym.

*D- wartości Licznika Adresów zostają z powrotem wyświetlane w postaci liczb szesnastkowych.

*E powoduje wyświetlenie trzech pustych linii na ekranie - komenda używana w celu wizualnego oddzielania bloków

*Hs wykonuje *E i wyświetla ciąg znaków s, który może ozna-

czać np. nagłówek modułu. Ciąg s jest wyświetlany po każdej następnej komendzie *E.

*S zatrzymuje listing asemblera. Po naciśnięciu dowolnego klawisza listing jest kontynuowany. *S nie zatrzymuje listingu na drukarce.

*F nazwa

Komenda *F umożliwia asemblację tekstu bezpośrednio z taśmy. Tekst jest wczytywany do bufora, po jednym bloku, a następnie tłumaczony. Pozwala to na tworzenie dużych programów, gdyż tekst programu nie zajmuje miejsca w pamięci.

Po literze F i spacji podaje się nazwę zbioru na taśmie. Nazwa może zawierać maksimum 10 znaków. Jeżeli nie podamy nazwy, to zostanie załadowany pierwszy zbiór znaleziony na taśmie.

Tekst przeznaczony do asemblacji z taśmy, musi być zapisany za pomocą komendy "T" edytora (nie "P") - "T" zapisuje tekst w porcjach (blokach) równych wielkości bufora z odpowiednimi przerwami, w czasie których tekst jest tłumaczony.

W programie GENS3 wielkość bufora równa się liczbie podanej w odpowiedzi na pytanie "Buffor size?" (patrz: rozdział 1) pomnożonej przez 256 bajtów. Zwiększenie wielkości bufora pozwala na skrócenie czasu asemblacji i ładowania tekstu kosztem zajętości pamięci i odwrotnie. Wielkość bufora w czasie nagrywania tekstu na taśmę musi być taka sama jak w czasie ładowania. Wygodnie jest więc nagrać tekst za pomocą komendy "P", a później, w razie potrzeby, nagrywać ten tekst w blokach (komendą "T") dla różnych wielkości buforów. Możliwość zmiany wielkości bufora okazała się dość żadko wykorzystywana przez użytkowników, dlatego autorzy następnych wersji GENS3 wprowadzili stałą wielkość bufora.

Po napotkaniu komendy "*F" pojawia się napis "start tape..."

(włącz taśmę), zarówno w pierwszym jak i w drugim przebiegu asemblera. Należy włączyć taśmę za każdym razem. Komunikat "Found nazwa" oznacza znalezienie zbioru o innej nazwie niż podana w komendzie *F a "Using nazwa" - ładowanie zbioru.

Przykład użycia komendy *F został podany w rozdziale 4.

ROZDZIAŁ 3. EDYTOR

3.1 Wprowadzenie

Edytor dołączony do wszystkich wersji programu GENS3 jest prostym edytorem wierszowym, zaprojektowanym do pracy ze wszystkimi systemami operacyjnymi Z80. Pozwala na szybką i wydajną edycję programów.

Aby zaoszczędzić ilość pamięci, w której przechowywany jest tekst programu użytkownika, spacje na początku każdego wiersza, między etykietą a mnemonikiem, między mnemonikiem a polem argumentów oraz między argumentami a komentarzem są zamieniane na jeden znak tabulatora. Spacje w komentarzach nie ulegają kompresji.

Uwaga: spacje nie mogą występować w etykietach, mnemonikach ani w polu argumentów.

W rozdziale 3 używa się następujących oznaczeń znaków sterujących:

ENTER		znak końca wiersza
EDIT	CAPS SHIFT 1	powrót do trybu przyjmowania komend
DELETE	CAPS SHIFT Ø	kasuje jeden znak na lewo od pozycji kursora
CS 8	CAPS SHIFT 8 (⇨)	przesuwa kursor do następnej pozycji tabulatora
CS 5	CAPS SHIFT 5 (⇦)	kasuje cały wiersz

Edytor zgłasza się automatycznie po uruchomieniu programu GENS3M21:

```
Copyright © HISOFT 1983,4
All Rights Reserved

>
```

Znak ">" oznacza gotowość edytora do przyjmowania komend o następującej postaci:

C N1,N2,S1,S2 ENTER

gdzie

C	symbol komendy (patrz: rozdział 3.2)
N1,N2	numery wierszy od 1 do 32767 włącznie
S1,S2	łańcuchy znaków o długości do 20 znaków

Przecinek używany jest do rozdzielania poszczególnych argumentów (patrz: komenda "S"). Spacje poza łańcuchami są pomijane. Argumenty nie są obowiązkowe, chociaż niektóre komendy (np. D - Delete) nie zostaną wykonane bez argumentów N1,N2. Edytor pamięta wartości argumentów ostatnio wprowadzonych przez użytkownika i używa ich, jeśli nowe wartości nie zostały podane. Zmienne N1 i N2 są początkowo ustawione na 10 a łańcuchy puste. Komendy nieważne są ignorowane i wyświetlany jest komunikat "Pardon?". Ten komunikat pojawia się również gdy długość łańcucha S2 przekracza 20 znaków; jeśli łańcuch S1 jest dłuższy niż 20 znaków to znaki od 21 wzwyż będą pomijane.
Komendy można wprowadzać wielkimi i małymi literami.
Podczas wpisywania komend można używać wszystkich znaków sterujących opisanych powyżej, np. CS 5 aby skasować wszystkie znaki.

3.2 Komendy edytora

Poniżej przedstawione zostały wszystkie komendy edytora. Gwiazdka (*) przy komendzie oznacza, że argumenty muszą być podane aby komenda została wykonana.

Wstawianie tekstu

Do programu można wstawić dowolny tekst piszęc numer wiersza, spację i żądany tekst lub za pomocą komendy I. Jeśli wprowadzony zostanie wiersz o numerze już istniejącym w tekście programu, to istniejący wiersz zostanie usunięty i zastąpiony nowym. Wpisanie samego numeru wiersza (bez tekstu) spowoduje usunięcie z programu wiersza o tym numerze (jeśli istnieje).
Wpisując tekst możemy używać wszystkich znaków sterujących wymienionych w rozdziale 3.1. Wiersz jest wprowadzany do programu po naciśnięciu ENTER. Tekst jest wprowadzany do bufora i jeśli bufor zostanie całkowicie zapełniony, nie będzie można wpisać żadnego znaku - należy wtedy zrobić miejsce za pomocą DELETE lub CS 5.
Jeśli podczas wpisywania tekstu, edytor stwierdzi, że koniec tekstu jest blisko RAMTOP, wyświetla komunikat: "Bad Memory!". Oznacza to, że nie można wpisać więcej tekstu i należy nagrać na taśmę część lub cały tekst.

Komenda: I n,m

Automatyczny tryb wstawiania. Edytor podaje numery wierszy zaczynając od n z krokiem m. Po pojawieniu się numeru, wpisujemy tekst (używając w razie potrzeby znaków sterujących), który kończymy znakiem ENTER. Wyjście z tego trybu następuje po naciśnięciu EDIT. Gdy numer wiersza przekroczy 32767, automatycznie nastąpi wyjście z trybu wstawiania.
Jeśli, pisząc tekst, dojdziemy do końca programu i bufor nie zostanie zapełniony (wielkość bufora wynosi 64 znaki),

to ekran zostanie przesunięty w górę umożliwiając dalsze pisanie. Tekst jest formatowany tak, aby numery wierszy były odzielone od tekstu.

Listowanie tekstu programu

Tekst można przeglądać przy użyciu komendy "L"; liczba linii ekranu wyświetlanych na raz jest z góry ustalona, ale można ją zmieniać za pomocą komendy "K".

Komenda: L n,m

Listowanie tekstu programu na ekranie od wiersza n do m włącznie. Gdy argumenty nie są podane, n zawsze równa się 1, a m = 32767, czyli wartości nie zależą od poprzednich. Cały program można więc wylistować używając komendy L bez argumentów. Linie są formatowane do lewego marginesu tak, że numery wierszy są wyraźnie widoczne. Automatyczna tabulacja tekstu powoduje oddzielenie różnych pól w linii. Po wyświetleniu ustalonej ilości linii (patrz: komenda "K") listing jest wstrzymywany aż do naciśnięcia dowolnego klawisza - oprócz EDIT, Który powoduje powrót do edytora.

Komenda: W n,m

Listowanie tekstu programu między wierszami n a m na drukarce. Pozostałe szczegóły identyczne jak w komendzie "L".

Komenda: K n

Ustala ilość linii ekranu wyświetlanych na raz, jak opisano w komendzie "L" powyżej.

Edycja (redagowanie) tekstu

Po wprowadzeniu pewnej ilości tekstu często zachodzi potrzeba edycji niektórych wierszy lub fragmentów programu.

Opisane poniżej komendy służą do dokonywania poprawek, usuwania, przesuwania i przenumerowywania wierszy.

Komenda: N n,m *

Użycie komendy "N" powoduje przenumerowanie wszystkich wierszy w programie. Pierwszy wiersz otrzymuje numer n, każdy następny o m większy od poprzedniego. Obydwa argumenty muszą być podane aby komenda została wykonana. Jeśli przekroczona zostanie wartość 32767, wówczas przywracana jest pierwotna numeracja.

Komenda: M n,m

Powoduje przepisanie treści wiersza o numerze n w wierszu o numerze m; zastępuje ewentualnie już istniejący wiersz m. Wiersz n pozostaje nie zmieniony. Jeśli wiersz o numerze n nie istnieje, komenda nie zostanie wykonana.

Komenda: D n,m *

Usuwa wiersze od numeru n do m włącznie z tekstu programu. Komenda zostanie wykonana gdy podane są obydwa argumenty, pod warunkiem, że n>m - ma to na celu uchronienie użytkownika przed pomyłką. Pojedynczy wiersz można usunąć z programu za pomocą komendy D, gdy n = m lub wpisując po prostu numer wiersza i ENTER.

Komenda: F n,m,f,s

Umożliwia odszukanie łańcuchów f w tekście programu między wierszami n a m i ewentualnie zastąpienie ich przez łańcuch s. Tekst między wierszami n a m jest przeglądany w poszukiwaniu łańcucha znaków f. Jeżeli taki łańcuch zostanie odnaleziony, wtedy wiersz, w którym znajduje się dany łańcuch jest wyświetlany (z kursorem umieszczonym na początku

łańcucha i następuje automatyczne wejście do trybu edycji. W trybie tym można zastąpić łańcuch f łańcuchem s lub korzystać z innych komend trybu edycji - patrz: komenda E.

Komenda: E n *

Przesyła wiersz o numerze n do edycji. Jeśli taki wiersz nie istnieje, komenda nie jest wykonywana, w przeciwnym wypadku wiersz ten jest kopiowany do bufora i wyświetlany na ekranie wraz z numerem. Poniżej jest wyświetlany ponownie numer wiersza i następuje wejście do trybu edycji. Wszelkie zmiany są dokonywane w buforze a nie na oryginalnym tekście, dlatego też w każdej chwili można przywrócić pierwotną treść wiersza.

W tym trybie możemy przesuwać kursor wzdłuż wiersza (rozpoczynając od pierwszego znaku) aby w odpowiednim miejscu wykonać jedną z poniższych komend trybu edycji:

" " (spacja)	przesuwa kursor o jedną pozycję w prawo (do następnego znaku); nie można przesunąć kursora poza koniec wiersza.
CS 8	przesuwa kursor do następnej pozycji tabulatora ale nie poza koniec wiersza.
DELETE	przesuwa kursor w lewo (do poprzedniego znaku), nie można przejść poza pierwszy znak w wierszu.
ENTER	kończy edycję wiersza z zachowaniem wszystkich dokonanych zmian.
Q	przerywa edycję, tzn. opuszcza tryb edycji nie uwzględniając żadnych dokonanych zmian i pozostawiając wiersz pierwotny.
R	ładuje do bufora oryginalny wiersz, ignorując dokonane zmiany i pozostaje w trybie edycji.

L — listuje pozostałą część wiersza znajdującego się w edycji (na prawo od bieżącej pozycji kursora), Wiersz pozostaje w trybie edycji z kursorem przesuniętym na początek wiersza.

K — Usuwa znak z aktualnej pozycji kursora.

Z — usuwą wszystkie znaki od aktualnej pozycji kursora do końca wiersza.

F — szuka następnago miejsca występowania łańcucha f zdefiniowanego uprzednio (patrz: komenda F). Powoduje automatyczne wyjście z trybu edycji (zachowując zmiany), o ile łańcuch f nie zostanie w tym wierszu odnaleziony. Jeżeli z kolei łańcuch ten zostanie odnaleziony w innym wierszu (w poprzednio określonym przedziale), nastąpi wejście do trybu edycji dla tego wiersza. Kursor jest zawsze umieszczany na początku odnalezionego łańcucha f.

S — aktualnie odnaleziony łańcuch f zostaje zastąpiony uprzednio zdefiniowanym łańcuchem s, po czym wykonywana jest komenda "F" trybu edycji. Komendy "F" i "S" stosowane są do przeszukiwania tekstu programu w celu odnalezienia wszystkich łańcuchów znaków f i ewentualnie zastąpienia ich łańcuchem s.

C — wejście w tryb zmiany znaków. Sygnalizowane jest to zmianą kursora na [+]. Wpisanie dowolnego znaku spwoduje zastąpienie znaku, na którym znajduje się kursor znakiem wpisanym i przesunięcie kursora o jedną pozycję. DELETE przesuwa kursor w lewo a CS 8 jest ignorowane. Po naciśnięciu ENTER nastąpi powrót do trybu edycji z kursorem umieszczonym za ostatnim zmienionym zakiem.

I — wejście w tryb wstawiania znaków. Kursor zmieni się na [*]. Wpisywane znaki są wstawiane na

bieżącą pozycję kursora. DELETE powoduje skasowanie jednego znaku na lewo od kursora, a CS 8 przesunięcie kursora do następnej pozycji tabulatora dopisując spacje. Podobnie jak w trybie zmiany, po naciśnięciu ENTER nastąpi powrót do trybu edycji z kursorem umieszczonym po ostatnio wpisanym znaku.

X Ustawia kursor na końcu wiersza z automatycznym wejściem w tryb wstawiania znaków (I).

Asemblacja i wykonywanie programu z edytora

Komenda: A

Wywołuje asembler GENS3M21. Szczegóły dotyczące asemblacji podane zostały w rozdziale 2.

Komenda: R

Jeśli asemblacja była bezbłędna oraz został podany adres wykonania za pomocą dyrektywy asemblera ENT (patrz: rozdział 2.7) wtedy komenda "R" może zostać użyta w celu wykonania kodu wynikowego. Jeżeli chcemy powrócić do edytora po wykonaniu kodu, należy umieścić w programie rozkaz RET (#C9). Oczywiście powrót taki będzie możliwy pod warunkiem, że stos w momencie wykonywania rozkazu RET będzie taki sam jak na początku programu. Dyrektywa RET nie działa, tzn. wykonanie kodu za pomocą komendy "R" nie jest możliwe, jeśli wybrana została opcja numer 16.

Współpraca z magnetofonem

Komenda: P n,m,s

Umożliwia zapisanie tekstu między wierszami n a m włącznie, na taśmie pod nazwą zdefiniowaną łańcuchem s. Przed wprowadzeniam komendy należy włączyć magnetofon - ładowanie rozpoczyna się z chwilą naciśnięcia ENTER.

Komenda: G,,s

Na taśmie odszukiwany jest zbiór o nazwie s, po czym jest on ładowany do pamięci komputera. Można również użyć komendy G w celu załadowania pierwszego napotkanego na taśmie zbioru. Po wprowadzeniu komendy pojawi się napis "Start tape ..." (włącz taśmę). Komunikat "Found nazwa" oznacza odszukanie zbioru o innej, niż podana, nazwie, a "Using nazwa" - ładowanie zbioru do pamięci. Błąd w trakcie ładowania sygnalizowany jest komunikatem "Tape error!".
Uwaga: jeśli w pamięci znajduje się już jakiś tekst, to program, który jest ładowany z taśmy zostanie do niego dołączony a całość przenumerowana poczynając od 1 z krokiem 1.

Komenda: T n,m,s

Powoduje zapis tekstu od wiersza n do m włącznie w postaci umożliwiającej późniejsze włączenie do innego programu za pomocą komendy asemblera *F (patrz: rozdział 2.9). Przed naciśnięciem ENTER należy włączyć taśmę.

Inne komendy

Komenda: B

Powrót do systemu operacyjnego ZX Spectrum. Szczegóły dotyczące ponownego wejścia do progrmau GENS3M21 oraz GENS3 omówiono w rozdziale 1.

Komenda: C

Dokonuje konwersji tekstów napisanych za pomocą edytora GENS1 na format używany przez GENS3M21. Tekst przeznaczony do konwersji należy załadować z taśmy za pomocą komendy "G".

Komenda: S,,d

Ta komenda pozwala zmienić separator oddzielający argumenty

w komendach edytora. Wstępnie, rolę separatora pełni ",".
Po tej komendzie separatorem będzie pierwszy znak łańcucha d; musi on być stosowany we wszystkich komendach (nawet w "S") dopóki nie zdefiniujemy nowego.

Spacja nie może być separatorem.

Przykład:

```
S,,!  nowym separatorem stanie się "!"
S!!,  z powrotem ","
```

Komenda: V

Wyświetla aktualne wartości argumentów N1,N2,S1 i S2.

Komenda: X

Wyświetla początkowy i końcowy adres tekstu programu.
Możemy wykorzystać te informacje aby nagrać tekst z Basica za pomocą zlecenia:

```
SAVE "nazwa" CODE ap,ak+1-ap
```

gdzie: ap - adres początkowy,
ak - adres końcowy.

Końcowy adres tekstu programu pozwoli nam też obliczyć ilość wolnej pamięci.

Edytor GENS3M21 może pracować z tekstem wpisanym przez użytkownika, nagranym z taśmy lub wygenerowanym w inny sposób, na przykład przez program MONS3M21 (patrz: instrukcja do programu MONS3M21, komenda "T", rozdział 2.2). W tym ostatnim przypadku, wygenerowany tekst należy umieścić od adresu początkowego podawanego przez komendę "X", a następnie wpisać adres końcowy (podany przez komendę "T" programu MONS3M21 lub obliczony na podstawie długości tekstu) do komórki TEXTEND o adresie "początek programu GENS3M21 + 54"; w końcu wejść do programu GENS3M21 via ciepły start (patrz: rozdział 1).

3.3 Przykład użycia edytora

Przypuśćmy, że napisaliśmy następujący program (używając komendy I10,10):

```
 10 *h   liczby losowe 16-bitowe
 20
 30 ; We - poprzednia liczba losowa w HL
 40 ; Wy - nowa liczba losowa w HL
 50
 60 LOS    PUSH AF      ; zachowaj rejestry
 70        PUSH BC
 80 PUSH HL
 90        ADD  HL,HL   ; *2
100        ADD  HL,HL   ; *4
110        ADD  HL,HL   ; *8
120        ADD  HL,HL   ; *16
130        ADD  HL,HL   ; *32
140        ADD  HL,HL   ; *64
150        ADD  HL,HL   ; *128
160        PIP  BC      ; poprzednia liczba
170        ADD  HL,DE
180        LD   DE,41
190        ADD  HL,DE
200        POP  BC
210        POP  AF
220        RER
```

W programie są następujące błędy:

wiersz 10 - komenda asemblera napisana małą literą

wiersz 80 - mnemonik w polu etykiet

wiersz 160 - "PIP" zamiast "POP"

wiersz 220 - "RER" zamiast "RET"

Poza tym powinien być dopisany dodatkowy wiersz z rozkazem ADD HL,HL po wierszu 150 oraz rejestry między wierszami 170 a 190 powinny zostać zamienione (BC zamiast DE).

Tekst programu możemy poprawić w sposób następujący:

E10 ENTER	następnie _ (spacja) C (tryb zmiany)
	H ENTER ENTER
E80 ENTER	I (tryb wstawiania) CS 8 (znak tabulatora)
	ENTER ENTER

155 CS 8 ADD CS 8 HL,HL CS 8 ; * 256 ENTER

E160 ENTER	CS 8 _ (spacja) C (tryb zmiany) 0
	ENTER ENTER
F170,190,DE,BC	S (zamiana łańcucha DE przez BC) S S
E190 ENTER	X __; *257+41 ENTER ENTER
	dopisaliśmy komentarz do wiersza 190
E220 ENTER	X DELETE T ENTER ENTER
N10,10	przenumerowujemy tekst

ROZDZIAŁ 4. PRZYKŁAD PRACY Z PROGRAMEM GENS3M21

Poniżej podany został typowy przykład pracy z programem GENS3M21. Zaleca się dokładne przestudiowanie tego przykładu użytkownikom, którzy nie pisali jeszcze programów w kodzie maszynowym lub nie są pewni jak korzystać z programu GENS3M21.

Przykład pokazuje proces kodowania programu mnożącego dwie liczby 16-bitowe, uruchamiania tego programu i zapisywania na taśmie przy użyciu komendy "T" tak, aby można go było włączać do innych programów. Tekst programu przed uruchomieniem powinien zostać nagrany na taśmie za pomocą komendy "P", aby nie trzeba go było przepisywać ponownie w razie przypadkowego zniszczenia.

Załóżmy, że program GENS3M21 został wczytany za pomocą programu ładującego "G/M 3M21", czyli od adresu 37000.

ETAP I - pisanie tekstu programu

Użyjemy komendy I edytora do wpisywania tekstu oraz ⇒(CS 8) przed mnemonikiem i między mnemonikiem a argumentami. Adresy pokazane w przykładzie służą jedynie jako ilustracja i mogą nie zgadzać się z otrzymanymi podczas asemblacji na konkretnej maszynie.

```
>I10,10 ENTER
 10 ; procedura mnozoca ENTER
 20 ; mnozy DE przez BC ENTER
 30 ; wynik umieszcza w DE_HL ENTER
 40 ENTER
 50 MNOZ    ENTER
 60         LD   HL,0 ENTER
 70         LD   A,16 ; licznik bitów ENTER
 80 PETLA   ADD  HL,HL ENTER
 90         EX   DE,HL ENTER
```

```
100         ADC  HL,HL ENTER
110         EX   DE,HL ENTER
120         JR   NC,NAST ENTER
130         ADD  HL,BC ENTER
140         JR   NC,NAS ENTER
150         INC  DE ENTER
160 NAST    DEC  A ; zmniejsz licznik bitow ENTER
170         JR   NZ,PETLA ENTER
180         RET ENTER
190 EDIT
```

```
>P10,180,MNOZENIE ENTER
```

Przed naciśnięciem ENTER w komendzie "P" należy włączyć taśmę.

ETAP II - uruchamianie programu

Najpierw sprawdzimy czy asemblacja przebiega prawidłowo. Wybierzemy opcje 2 i 4 aby asembler nie generował kodu wynikowego i nie wyświetlał listingu.

```
>A ENTER
Table size: ENTER
Options: 6 ENTER

*HISOFT GENS3M2 ASSEMBLER*
       ZX SPECTRUM
Copyright (C) HISOFT 1983,4
All rights reserved

Pass 1 errors: 00

Pass 2 errors: 00
*WARNING* NAS absent
Table used :   47 from  131
>
```

Jak widać, popełniliśmy błąd w wierszu 140 pisząc NAS zamiast NAST. Poprawiamy błąd:

```
>E140 ENTER

  140         JR   NC,NAS
  140 [C]
```

naciskamy X

```
  140        JR   NC,NAS■
```

i wpisujemy brakującą literę T. Po dwukrotnym naciśnięciu ENTER poprawiony wiersz zostanie wstawiony do programu. Kolejna asemblacja powinna być bezbłędna.

Napiszemy teraz program testujący naszą procedurę:

>N1000,10 ENTER (przenumerowujemy tekst zaczynając od numeru 1000, aby wpisać nowy tekst na początku)

```
>I10,10 ENTER
 10 ; program testujący ENTER
 20 ; procedure mnozenia ENTER
 30 ENTER
 40        LD   BC,(MNOZNA) ENTER
 50        LD   DE,(MNOZNIK) ENTER
 60        CALL MNOZ ENTER
 70        LD   A,D ENTER
 80        CALL PISZ_A ENTER
 90        LD   A,E ENTER
100        CALL PISZ_A ENTER
110        LD   A,H ENTER
120        CALL PISZ_A ENTER
130        LD   A,L ENTER
140        CALL PISZ_A ENTER
150        RET ENTER
160 ENTER
170 PISZ_A PUSH AF ENTER
180        RRCA ENTER
190        RRCA ENTER
200        RRCA ENTER
210        RRCA ENTER
220        CALL DALEJ ENTER
230        POP  AF ENTER
240 DALEJ  AND  %1111 ENTER
250        ADD  A,#90 ENTER
260        DAA ENTER
```

```
270          ADC  A,#40 ENTER
280          DAA ENTER
290          LD   IY,#5C3A ENTER
300          RSR  16 ENTER
310          RET ENTER
320 MNOZNA DEFW 1000 ENTER
330 MNOZNI DEFW 50 ENTER
340 *E ENTER
350 EDIT
>
```

Teraz możemy przeprowadzić asemblację procedury testującej i mnożącej łącznie:

```
>A ENTER

Table size: ENTER
Options: 6 ENTER

*HISOFT GENS3M2 ASSEMBLER*
      ZX SPECTRUM

Copyright (C) HISOFT 1983,4
All rights reserved

B69F  300         RSR  16
*ERROR* 02                    (nacisnij dowolny klawisz)

Pass 1 errors: 01

Table used:   98 from  101
>
```

W pierwszym przebiegu asemblera został wykryty jeden błąd - powinno być RST zamiast RSR. Poprawiamy błąd:

```
>E300 ENTER

  300         RSR  16
  300 [C]
```

Przesuwamy kursor (klawisz SPACE) na literę "R" i naciskamy C - tryb zmiany znaków

```
  300         RS[+] T ENTER ENTER
>
```

Dokonujemy ponownie asemblacji i jeśli jest prawidłowa, możemy wykonać nasz program. Przedtem jednak musimy wskazać miejsce

(adres), od którego edytor ma zacząć wykonywać kod. Wpisujemy wiersz:

35 ENT $ ENTER

Teraz, po asemblacji (w odpowiedzi na Options: naciskamy ENTER), wyświetlony zostanie dodatkowy komunikat:

Executes: 46718

Wykonujemy program za pomocą komendy "R":

>R ENTER

0000E15A>

#E15A równa się 50000 (1000* 50) - program możemy przetestować również dla innych liczb. Wystarczy wpisać inne wartości w wierszach 320 i 330. Przed wykonaniem programu, gdy w tekście zostały zrobione jakiekolwiek zmiany, należy przeprowadzić ponowną asemblację.

Kolejną czynnością będzie nagranie procedury na taśmę tak, aby można ją było włączać do innych programów:

>T1000,1500,MNOZ ENTER

Przed naciśnięciem ENTER należy włączyć taśmę. Tak nagraną procedurę możemy wykorzystywać w innych programach w następujący sposób, np.:

```
500 ; włącz procedurę MNOZ
510 *F MNOZ
520 ; następne procedury
```

Gdy podczas asemblacji pojawi się napis "Start tape...", należy włączyć taśmę z nagraną procedurą MNOZ. Czynność tę należy powtórzyć również w drugim przebiegu asemblera.

ZAŁĄCZNIK 1. OPIS BŁĘDÓW

ERROR 1	błąd w składni danego wiersza
ERROR 2	nierozpoznany mnemonik
ERROR 3	nieprawidłowo sformułowany rozkaz
ERROR 4	symbol zdefiniowany więcej niż jeden raz
ERROR 5	ten wiersz zawiera nieważny znak, tzn. znak, który nie jest ważny w tym kontekście
ERROR 6	jeden z argumentów rozkazu jest nieprawidłowy
ERROR 7	symbol w tym wierszu jest słowem zastrzeżonym
ERROR 8	niedopasowane rejestry
ERROR 9	zbyt wiele rejestrów w tym wierszu
ERROR 10	wartość wyrażenia, które powinno być 8-bitowe, ma więcej niż 8 bitów
ERROR 11	rozkazy JP IX+n oraz JP IY+n są nieważne
ERROR 12	błąd w sformułowaniu dyrektywy asemblera
ERROR 13	nieprawidłowe odniesienie w przód, tzn. wyrażenie w EQU zawiera niezdefiniowany symbol
ERROR 14	dzielenie przez zero
ERROR 15	nadmiar w operacji mnożenia
Bad Org!	wartość wyrażenia ORG jest taka, że generowany kod wynikowy mógłby spowodować zniszczenie programu GENS3M21, tekstu programu użytkownika lub tablicy symboli. Sterowanie przekazywane jest do edytora
Bad Memory!	nie ma miejsca w pamięci na więcej tekstu programu. Należy zapisać na taśmę cały tekst lub jego część
Tape error!	błąd we wczytywaniu z taśmy
No Table Space!	za mały obszar pamięci został przeznaczony na Tablicę Symboli (w odpowiedzi na "Table size:"). Sterowanie przekazywane jest do edytora

ZAŁĄCZNIK 2. SŁOWA ZASTRZEŻONE, MNEMONIKI, DYREKTYWY I KOMENDY ASEMBLERA

Słowa zastrzeżone nie mogą być używane jako etykiety ale mogą stanowić część etykiety. Wszystkie słowa zastrzeżone są pisane wielkimi literami.

Lista słów zastrzeżonych:

A	B	C	D	E	H	L	R

AF	AF´	BC	DE	HL
IX	IY	SP	Z	NZ
M	P	PE	PO	

Lista mnemoników Z80:

ADC	ADD	AND	BIT	CAL	CCF	CP	CPD	CPDR
CPI	CPIR	CPL	DAA	DEC	DI	DJNZ	EI	EX
EXX	HALT	IM	IN	INC	IND	INDR	INI	INIR
JP	JR	LD	LDD	LDDR	LDI	LDIR	NEG	NOP
OR	OTDR	OTIR	OUT	OUTD	OUTI	POP	PUSH	RES
RET	RETI	RETN	RL	RLA	RLC	RLCA	RLD	RR
RRA	RRC	RRCA	RRD	RST	SBC	SCF	SET	SLA
SRA	SRL	SUB	XOR					

Lista dyrektyw asemblera:

DEFB	DEFM	DEFS	DEFW	ELSE
END	ENT	EQU	IF	ORG

Lista komend asemblera:

*D	*E	*H	*L	*S	*C
*F					

Część II

M O N S 3 M 2 1

MONITOR I DEBUGGER

ROZDZIAŁ 1. URUCHOMIENIE PROGRAMU

MONS3M21 można załadować w dowolne miejsce w pamięci. Długość programu wynosi 6068 bajtów (MONS3 - 5760), łącznie z własnym, wewnętrznym stosem. Jeśli użyjemy oryginalnego programu ładującego "G/M 3M21", MONS3M21 zostanie załadowany od adresu 30000. Chcąc umieścić program w innym miejscu (np. pod koniec pamięci), należy wprowadzić następujące komendy:

```
CLEAR 59467
LOAD "MONS3M21" CODE 59468
RANDOMIZE USR 59468
```

Ostatnia komenda to skok do adresu 59468, czyli do miejsca, od którego program został załadowany. Skok do tego miejsca wykonujemy zawsze, gdy chcemy uruchomić program ponownie, tzn. po powrocie do systemu operacyjnego ZX Spectrum.

Program MONS3 ładujemy i uruchamiamy w ten sam sposób, ale ponowne wejście do programu może nastąpić, poprzez skok do adresu o 29 większego. Na przykład, po załadowaniu MONS3 od adresu 40000, piszemy:

RANDOMIZE USR 40000 za pierwszym razem
RANDOMIZE USR 40029 za każdym następnym razem, gdy chcemy uruchomić program.

Po wejściu do programu MONS3M21 ukazuje się na kilka sekund napis "MONS3M2 Debugger © Hisoft 1983,4", a następnie tak zwana "tablica MONS3" (patrz: załącznik "Tablica MONS3"). Na tablicy ukazane są zawartości wszystkich rejestrów oraz znaczniki stanu mikroprocesora Z80; 24 bajty pamięci wokół Wskaźnika Pamięci (oznaczonego na tablicy znakami: > <) oraz zapis symboliczny rozkazu, adresowanego przez Wskaźnik Pamięci.

Adresy pamięci, zawartości rejestrów oraz komórek pamięci są zapisane w systemie szesnastkowym. System zapisu adresów pamięci oraz zawartości rejestrów (oprócz AF i IR) możemy zamienić na dziesiętny za pomocą komendy # (SYMBOL SHIFT 3). Komendy wprowadzamy w odpowiedzi na znak ">", znajdujący się w dolnej części ekranu. Komendy można wprowadzać małymi lub wielkimi literami, niektóre z nich wymagają użycia CAPS SHIFT (CS) lub SYMBOL SHIFT (SS).

Liczby podawane w komendach (oprócz komendy H) muszą być w systemie szesnastkowym. Możemy wprowadzić dowolną ilość cyfr szesnastkowych (Ø-9, A-F lub a-f) i zakończyć ten ciąg dowolnym innym znakiem (DELETE kasuje ostatnią cyfrę).

Jeśli znakiem kończącym jest "-" (minus), to liczba jest traktowana jako ujemna - w kodzie U2 (np. 1800- równa się E800).

Tylko ostatnio wpisane cztery cyfry są wyświetlane na ekranie.

Komendy wykonywane są natychmiast - nie potrzeba naciskać ENTER. Po wykonaniu każdej komendy, cała tablica jest uaktualniana. Nieważne komendy są ignorowane.

W dowolnym momencie można wrócić do systemu operacyjnego ZX Spectrum naciskając EDIT (CAPS SHIFT 1).

UWAGA: MONS3M21 wyłącza przerwania aby zapewnić prawidłową pracę - użytkownik powinien uważać, aby nie włączać przerwań (rozkazem EI) podczas pracy z programem.

ROZDZIAŁ 2. KOMENDY

2.1 Ustawianie Wskaźnika Pamięci

M

ustawia Wskaźnik Pamięci na podany adres.

Adres wprowadzamy w systemie szesnastkowym w odpowiedzi na ":", np.:

M:E43A

ENTER

zwiększa Wskaźnik Pamięci o jeden

CAPS SHIFT 8 lub SYMBOL SHIFT 2

zwiększa Wskaźnik Pamięci o osiem

CAPS SHIFT 7

zmniejsza Wskaźnik Pamięci o jeden

CAPS SHIFT 5

zmniejsza Wskaźnik Pamięci o osiem

X

ustawia Wskaźnik Pamięci na adres skoku bezwzględnego (JP lub CALL)

Komenda X ustawia Wskaźnik Pamięci na adres wskazywany przez Wskaźnik Pamięci (mniej znaczący bajt) i Wskaźnik Pamięci + 1 (bardziej znaczący bajt). Patrz przykład w komendzie V.

V

używana w połączeniu z komendą X

V powoduje ustawienie Wskaźnika Pamięci na adres sprzed ostat-

nio wykonanej komendy X. Przykład użycia komend X i Y:

```
 COF2  B8
>COF3  CC <        CALL Z,#CA47
 COF4  47
 COF5  CA
 COF6  19
```

W adresie COF3, wskazywanym przez Wskaźnik Pamięci, znajduje się rozkaz CALL Z,#CA47. Przesuwamy Wskaźnik o jeden (komenda ENTER) tak, aby wskazywał na adres podprogramu:

```
 COF2  B8
 COF3  CC
>COF4  47 <
 COF5  CA
 COF6  19
```

i naciskamy X:

```
 CA46  C9
>CA47  01 <
 CA48  EA
 CA49  FF
 CA50  21
```

Wskaźnik Pamięci jest ustawiony na adres CA47. Po obejrzeniu podprogramu, możemy wrócić do programu głównego naciskając V - - Wskaźnik Pamięci będzie wskazywał adres COF4.

Za pomocą komendy V można wrócić do adresu sprzed ostatnio wykonanej komendy X; wszystkie wcześniejsze adresy, z których zostały wykonane komendy X są wymazane z pamięci programu MONS3.

O

ustawia Wskaźnik Pamięci na adres skoku względnego (JR lub DJNZ).

Wykonanie komendy O polega na dodaniu do wartości Wskaźnika

Pamięci zawartości komórki pamięci wskazywanej przez Wskaźnik, umieszczeniu wyniku we Wskaźniku Pamięci oraz odpowiedniej zmianie tablicy MONS3. Patrz: przykład w komendzie U.

U

używana w połączeniu z komendą O.

U powoduje ustawienie Wskaźnika Pamięci na adres sprzed ostatnio wykonanej komendy O. Przykład użycia komend O i U:

```
  DA49  0D
> DA4A  28 <          JR  NZ,#DA4E
  DA4B  02
  DA4C  E1
  DA4D  C9
```

W adresie DA4A, wskazywanym przez Wskaźnik Pamięci, znajduje się rozkaz JR NZ,#DA4E. Przesuwamy Wskaźnik o jeden (ENTER), tak, aby wskazywał wielkość przesunięcia względnego (#02):

```
  DA49  0D
  DA4A  28
> DA4B  02 <
  DA4C  E1
  DA4D  C9
  DA4E  29
  DA4F  AC
  DA50  1F
```

Po naciśnięciu O do adresu we Wskaźniku Pamięci (#DA4B), zostanie dodana zawartość komórki adresowanej przez Wskaźnik (#02) oraz 1.

```
  DA4D  C9
> DA4E  29 <
  DA4F  AC
  DA50  1F
```

Za pomocą komendy U możemy wrócić do adresu sprzed ostatnio

wykonanej komendy O. (w powyższym przykładzie #DA4B).

, (przecinek)

Ustawia Wskaźnik Pamięci na adres znajdujący się na stosie (wskazywany przez wskaźnik stosu - SP).

Komenda jest stosowana, gdy chcemy poznać zawartość pamięci wokół adresu powrotu z podprogramu. Komenda ta jest często używana w połączeniu z komendą SS K.

2.2 Deasemblacja kodu

SYMBOL SHIFT 4 $

dokonuje deasemblacji kodu od adresu wskazywanego przez Wskaźnik Pamięci.

Po naciśnięciu SS 4 tablica MONS3 zostaje zastąpiona wydrukiem o następującej postaci:

adres	zawartość komórek pamięci tworząca 1 rozkaz	symboliczny zapis rozkazu

Po naciśnięciu dowolnego klawisza zostaną wyświetlone następne rozkazy, zajmujące około 20 bajtów pamięci. Ponowne naciśnięcie **SS 4 spowoduje** powrót do tablicy MONS3. Zobacz również: komenda SS 3 w rozdziale 2.8.

T

dokonuje deasemblacji bloku kodu z możliwością wyprowadzenia wydruku na drukarkę oraz wygenerowania tekstu w formie umożliwiającej używanie go przez edytor programu GENS3M21.

Adresy początku i końca bloku podajemy w odpowiedzi na

"First:" (adres pierwszego bajtu) i "Last:" (adres ostatniego bajtu). Komenda nie zostanie wykonana jeśli adres początku jest większy od adresu końca. Po wpisaniu adresów pojawi się pytanie "Printer?" (drukarka) - należy odpowiedzieć "Y" (tylko wielka litera), aby skierować tekst na drukarkę. Następnie wyświetlony zostanie napis "Text:", po którym wprowadzamy adres początkowy pod jakim chcemy umieścić tekst deasemblacji. Wciśnięcie ENTER bez adresu spowoduje, że tekst nie zostanie wygenerowany. Jeśli wpiszemy adres, tekst deasemblacji, w formie umożliwiającej używanie go przez edytor programu GENS3M21, zostanie umieszczony począwszy od tego adresu. Użycie tekstu w programie GENS3M21 będzie możliwe, jeśli tekst zostanie wygenerowany lub przesunięty (patrz: komenda I) do adresu podanego przez komendę "X" edytora GENS3M21 (tzn. tam, gdzie edytor umieszcza tekst programu użytkownika).

Po wygenerowaniu tekstu we właściwym miejscu a przed wejściem do programu GENS3M21 należy wpisać adres końca tekstu do komórki TEXTEND, znajdującej się w programie GENS3M21 (patrz: instrukcja do programu GENS3M21, rozdział 3.2). Adres końca tekstu zostanie podany po wykonaniu komendy T w komunikacie "Text end=" (patrz niżej).

MONS3M21 sprawdza czy generowany tekst nie zachodzi na program MONS3M21; w przypadku zaistnienia takiego niebezpieczeństwa, wykonywanie komendy T jest przerywane - po naciśnięciu dowolnego klawisza nastąpi powrót do tablicy MONS3.

Po podaniu adresu w odpowiedzi na "Text:", pojawi się napis "Workspace:", po którym należy podać adres początku wolnego obszaru w pamięci, który zostanie użyty jako tablica symboli. Potrzebna wielkość pamięci wynosi 2 bajty razy ilość

etykiet generowanych podczas deasemblacji (patrz niżej). Jeśli wciśniemy ENTER, tablica symboli będzie tworzona od adresu #6000 (24576).
Jeśli w bloku kodu przeznaczonego do deasemblacji znajdują się obszary danych (nie zawierające rozkazów Z80), to należy podać adresy początku i końca każdego z tych obszarów, w odpowiedzina "First:" i "Last:", a po zdefiniowaniu wszystkich, wcisnąć dwa razy ENTER. Dla obszarów danych zostaną wygenerowane dyrektywy asemblera DEFB (patrz: instrukcja do programu GENS3M21 - rozdział 2.7) oraz odpowiednie znaki w kodzie ASCII. Ponieważ adresy definiowanych obszarów są przechowywane na końcu programu MONS3M21, możemy zdefiniować tyle obszarów danych, ile jest wolnej pamięci w programie MONS3M21, Uwaga: komenda T niszczy wszystkie uprzednio wpisane adresy powrotu - patrz: komenda W.
Gdy wszystkie dane zostaną wpisane, następuje wymazanie ekranu i po chwili przerwy, w czasie której generowana jest tablica symboli, na ekranie lub drukarce ukazuje się listing deasemblacji. Postać listingu jest taka jak w komendzie SS 4, z tym, że dołączone jest pole etykiet. Listing można zatrzymać naciskając ENTER lub SPACE, po czym naciśnięcie:

CS 5 spowoduje powrót do tablicy MONS3

a innego klawisza - kontynuację listingu

Jeśli zostanie napotkany nieważny format rozkazu, jest on tłumaczony jako NOP, a zawartość bajtu oznaczona znakiem "*".
Po zakończeniu deasembliacji, jeśli żądaliśmy generowania tekstu, wyświetlony zostanie komunikat "Text end = aa", gdzie aa jest adresem końca tekstu (dziesiętnie lub szesnastkowo), który należy wpisać do komórki TEXTEND w programie GENS3M21.

Powrót do tablicy MONS3 nastąpi po naciśnięciu dowolnego klawisza.

Podczas deasemblacji tworzone są etykiety postaci Lxxxx, gdzie "xxxx" jest adresem w zapisie szesnastkowym. Etykiety są tworzone tylko dla adresów leżących w granicach podanego obszaru, inne adresy są podawane w postaci liczb dziesiętnych lub szesnastkowych. Na przykład, jeśli dokonujemy deasemblacji kodu w granicach od #7000 do #8000, wtedy rozkaz o kodzie DAF574 zostanie zapisany w postaci: JP C,L74F5, a np. rozkaz CC27E0 - jako CALL Z,#E027 lub CALL Z,57383, jeśli adresy są wyświetlane w zapisie dziesiętnym. Odpowiednie adresy, jeśli jest odwołanie do nich, zostaną opatrzone etykietą (w polu etykiet, przed mnemonikiem), tylko w przypadku generowania tekstu programu.

Przykład użycia komendy T.

Załóżmy, że program GENS3M21 jest załadowany od adresu 37000, tak więc adres pod którym edytor umieszcza tekst programu (uzyskany za pomocą komendy X edytora) wynosi 45866 (#B32A)

T

First: FFDF	adres początku bloku kodu przeznaczonego do deasemblacji
Last: FFFE	adres końca bloku
Printer? ENTER	tekst zostanie wyświetlony na ekranie
Text: B32A	adres początkowy, pod jakim chcemy umieścić tekst deasemblacji
Work space: E000	adres początku wolnego obszaru w pamięci przeznaczonego na tablicę symboli
First: FFF9	adres początkowy obszaru nie zawierającego rozkazów Z80
Last: FFFE	adres końcowy tego obszaru
First: ENTER	koniec definiowania
Last: ENTER	

```
FFDF  A7                       AND  A
FFE0  21 30 5C                 LD   HL,#3C30
FFE3  ED 5B 4F 5C              LD   DE,(#5C4F)
FFE7  ED 52                    SBC  HL,DE
FFE9  23                       INC  HL
FFEA  22 1A 5C                 LD   (#5C1A),HL
FFED  11 30 5C                 LD   DE,#5C30
FFF0  21 F9 FF                 LD   HL,LFFF9
FFF3  01 06 00                 LD   BC,#0006
FFF6  ED B0                    LDIR
FFF8  C9                       RET
FFF9  56 FE C4       LFFF9     DEFB "V",#FE,#C4
FFFC  15 53 80                 DEFB #15,"S",#80
```

Text end = #B3E4

Adres końca tekstu, #B3E4, musimy wpisać do komórki TEXTEND programu GENS3M21, która znajduje się pod adresem 37054.

Zamieniamy ten adres na postać szesnastkową (patrz: komenda H):

H:37054 = 90BE

i ustawiamy Wskaźnik Pamięci na ten adres:

M:90BE

Możemy teraz wpisać adres końca tekstu, #B3E4, pamiętając aby mniej znaczący bajt wpisać najpierw:

E4 ENTER	E4 zostanie wpisane do adresu #90BE i Wskaźnik Pamięci przesunie się o 1 (patrz: rozdział 2.6)
B3 SPACE	wpisujemy B3

Po wpisaniu adresu końca tekstu możemy wejść do programu GENS3M21 via ciepły start (patrz instrukcja do programu GENS3M21 rozdział 1).

2.3 Wykonywanie programów w kodzie wynikowym

SYMBOL SHIFT Z

wykonuje jeden rozkaz programu

Przed użyciem komendy SS Z zarówno licznik rozkazów PC, jak i Wskaźnik Pamięci muszą być ustawione na adres rozkazu, który zamierzamy wykonać (patrz: rozdział 2.7). Np. :

M:782D	ustawiamy Wskaźnik Pamięci na adres #782D
782D.	wpisujemy #782D do rejestru PC

SS Z wykonuje bieżący rozkaz i uaktualnia tablicę MONS3. Za pomocą komendy SS Z możemy wykonywać dowolny program w RAM i ROM pod warunkiem, że nigdzie nie występuje rozkaz włączający przerwania (EI).

W

wpisuje adres powrotu w miejscu wskazywanym przez Wskaźnik Pamięci.

Jest to komenda pomocnicza, używana najczęściej w połączeniu z komendą SS K (patrz niżej).
Adres powrotu to po prostu rozkaz CALL do podprogramu zajmującego się wyświetlaniem tablicy MONS3. Wpisanie adresu powrotu umożliwia zatrzymanie wykonywania programu użytkownika w żądanym miejscu, w celu obejrzenia zawartości rejestrów, stanu znaczników itd. Na przykład, jeśli chcemy zatrzymać wykonywanie programu w adresie #E84E, to należy ustawić Wskaźnik Pamięci na ten adres i nacisnąć W aby wstawić adres powrotu w tym miejscu.
3 bajty kodu, które znajdują się w adresach #E84E, #E84F, #E850, zostaną zapisane w pamięci na końcu programu MONS3M21,

a na ich miejsce będzie wpisany rozkaz CALL z odpowiednim adresem. Gdy, podczas wykonywania programu użytkownika, zostanie napotkany powyższy rozkaz, oryginalne 3 bajty będą wpisane na swoje miejsce i zostanie wyświetlona tablica MONS3 ze wszystkimi rejestrami i znacznikami sprzed momentu wykonania rozkazu CALL.

Program MONS3M21 używa obszaru pamięci znajdującego się na końcu programu, do przechowywania informacji o wpisanych adresach powrotu. Oznacza to, że możemy wpisać tyle adresów powrotu ile jest dostępnej pamięci; na każde wykonanie komendy W potrzebne jest 5 bajtów pamięci. Po wykonaniu rozkazu CALL z adresem powrotu, automatycznie odtwarzana jest zawartość pamięci jaka była przed wpisaniem adresu powrotu.

Uwaga: wykonanie komendy T, która używa tego samego obszaru pamięci, spowoduje utratę zawartości tych komórek pamięci, w które zostały uprzednio wpisane adresy powrotu.

Adresy powrotu mogą być wpisywane tylko do pamięci RAM. Ponieważ adres powrotu składa się z trzy-bajtowego rozkazu CALL, należy zwrócić szczególną uwagę na miejsce jego wpisania. Na przykład, w poniższym fragmencie:

```
A400  1A        LD  A,(DE)
A401  FE        CP  A,#F0
A402  F0
A403  28        JR  Z,#A406
A404  01
A405  AF        XOR A
A406  C3        JP  #43A3
A407  A3
A408  43
```

Jeśli wpiszemy adres powrotu w #A405 i następnie zaczniemy wykonywać program od adresu A400, to do rejestru A zostanie

Wpisana zawartość komórki pamięci adresowanej przez parę rejestrów DE. Jeżeli w wyniku porównania, znacznik Z zostanie ustawiony na 1, to wykonany będzie rozkaz skoku względnego do adresu #A406. Ale w komórce o adresie #A406 znajduje się teraz mniej znaczący bajt adresu rozkazu CALL; kod programu został zmieniony i może spowodować nieprzewidziane skutki. Taka sytuacja występuje stosunkowo rzadko, niemniej jednak należy się przed nią zabezpieczyć - najprostszym sposobem jest wykonywanie programu za pomocą komendy SS Z.

SYMBOL SHIFT K +

kontynuuje wykonywanie programu od adresu znajdującego się w liczniku rozkazów (PC).

Komendy tej używa się najczęściej w połączeniu z komendą W. Najlepiej wyjaśni to przykład:

Wykonujemy poniższy program po jednym rozkazie (używając SS Z) i doszliśmy do adresu #C3A2.

```
C3A2  ED 4A     SBC   HL,BC
C3A4  A9        XOR   C
C3A5  1E FF     LD    E,#FF
C3A7  80        ADD   A,B
C3A8  BE        CP    A,(HL)
C3A9  D4 21 D0  CALL  NC,#D021
C3AC  11 A0 A0  LD    DE,#A0A0
C3AF  CD 11 C4  CALL  #C411
C3B2  DA E2 C3  JP    C,#C3E2
```

Interesuje nas stan znaczników po powrocie z podprogramu zaczynającego się od adresu #C411. Ustawiamy Wskaźnik Pamięci na adres #C3B2 (po rozkazie CALL #C411) i wpisujemy adres powrotu używając komendy W. Następnie naciskamy SYMBOL SHIFT i K.

Program zostanie wykonany od adresu znajdującego się w liczniku rozkazów (PC), czyli, w tym przypadku, od #C3A2. Po dojściu do adresu, w który wpisaliśmy adres powrotu (#C3B2), wykonanie programu zostanie zatrzymane, po czym tablica zostanie uaktualniona (w programie MONS3).W programie MONS3M21 po wykonaniu komendy SS K generowany jest krótki sygnał dźwiękowy a tablica jest uaktualniana po naciśnięciu dowolnego klawisza.

SYMBOL SHIFT T >

wpisuje adres powrotu po bieżącym rozkazie i kontynuuje wykonywanie programu.

Przykład:

```
B231 2B        DEC   HL
B232 CD        CALL  #A782
B233 82
B234 A7
B235 21        LD    HL,#9FFF
B236 FF
B237 9F
```

Wykonując program po jednym rozkazie doszliśmy do adresu #B232. Jeżeli teraz użyjemy komendy SS Z, to wykonywanie programu będzie kontynuowane od adresu #A782 - początek podprogramu. Jeśli nie interesuje nas przeglądanie tego podprogramu, to wystarczy nacisnąć SYMBOL SHIFT i T. Adres powrotu zostanie wpisany po rozkazie CALL #A782, czyli w adresie #B235, i wykonywanie programu będzie kontynuowane podobnie jak w przypadku komendy SS K.

J

powoduje wykonanie programu od podanego adresu.

Adres wpisujemy po ":". Przed skokiem do podanego adresu nastąpi wyczyszczenie stosu wewnętrznego. Jeśli chcemy, aby po wykonaniu programu, nastąpił powrót do tablicy MONS3, należy wpisać adres powrotu (W) w żądanym miejscu.

Przykład:

J:C000 ENTER wykona program od adresu #C000

Przed wpisaniem znaku kończącego adres możemy wycofać się z tej komendy naciskając CAPS SHIFT 5.

Uwaga: komenda J niszczy zawartość rejestrów przed wykonaniem danego programu. Program do wykonania nie powinien więc zakładać określonych wartości w rejestrach. Jeśli chcemy, aby program został wykonany z danymi wartościami rejestrów, należy użyć komendy SS K.

2.4 Wyświetlanie zawartości pamięci.

L

wyświetla blok pamięci od adresu we Wskaźniku Pamięci

L usuwa tablicę MONS3 i wyświetla zawartość 80 komórek pamięci poczynając od adresu wskazywanego przez Wskaźnik Pamięci. Obraz składa się z dwudziestu linii, każda zawiera adres (dziesiętny lub szesnastkowy - tak, jak na tablicy - patrz: komenda SS 3), 4 liczby szesnastkowe oznaczające zawartości kolejnych czterech bajtów pamięci oraz 4 znaki w kodzie ASCII odpowiadające tym bajtom. Jeżeli zawartość danego bajtu przekracza liczbę 127 (#7F), zostaje ona pomniejszona o 128, dla celów reprezentacji w kodzie ASCII. Wartości od Ø do 31 włącznie są przedstawiane jako ".".

Powrót do tablicy MONS3 nastąpi po naciśnięciu CAPS SHIFT 5;

naciśnięcie dowolnego innego klawisza (oprócz CAPS SHIFT 1) spowoduje wyświetlenie następnego bloku pamięci.

SYMBOL SHIFT P "

jest to taka sama komenda jak L z tym, że wydruk skierowany jest na drukarkę, a nie na ekran.

Powrót do tablicy MONS3, po wydrukowaniu jednej strony (80 bajtów pamięci), następuje po naciśnięciu CAPS SHIFT 5, a naciśnięcie dowolnego innego klawisza (oprócz CAPS SHIFT 1) spowoduje wydrukowanie kolejnej strony.

2.5 Przeszukiwanie pamięci

G

przegląda pamięć w poszukiwaniu podanego ciągu znaków.

Znaki podajemy w odpowiedzi na ":", w postaci liczb szesnastkowych (po jednym bajcie) zakończonych ENTER. Wciśnięcie ENTER, bez liczby, zakończy definiowanie poszukiwanego ciągu.

Przeszukiwanie rozpoczyna się od adresu we Wskaźniku Pamięci. Z chwilą znalezienia podanego ciągu, tablica zostaje zmieniona tak,że Wskaźnik Pamięci jest ustawiony na pierwszy znak szukanego ciągu.

Przykład: chcemy znaleźć ciąg #2B #02 (2 bajty), zaczynając od adresu #C250:

M:C250 ENTER	ustawiamy Wskaźnik Pamięci na #C250
G:2B ENTER	definiujemy pierwszy bajt ciągu
:2 ENTER	drugi bajt
: ENTER	koniec definicji

Patrz: komenda N.

N

przegląda pamięć w poszukiwaniu ostatnio zdefiniowanego ciągu (patrz: komenda G)

Komenda G pozwala na zdefiniowanie poszukiwanego ciągu i wskazuje pierwszy napotkany ciąg - dalsze przeszukiwanie umożliwia komenda N. N zaczyna przeszukiwać pamięć od adresu we Wskaźniku Pamięci i ustawia Wskaźnik na adres pierwszego znaku znalezionego ciągu.

2.6 Modyfikacja zawartości pamięci

Zawartość komórki pamięci o adresie wskazywanym przez Wskaźnik Pamięci możemy zmieniać wpisując liczbę szesnastkową zakończoną dowolnym znakiem (patrz: rozdział 1). Dwie mniej znaczące cyfry (jeżeli napiszemy tylko jedną cyfrę, zostanie ona uzupełniona zerem z lewej strony) zostaną wpisane do komórki pamięci o adresie we Wskaźniku Pamięci, a następnie, jeśli znakiem kończącym był symbol którejś z komend, komenda ta zostanie wykonana.

Przykłady:

3C SPACE	#3C zostanie wpisane do komórki adresowanej przez Wskaźnik Pamięci
D ENTER	#0D zostanie wpisane i Wskaźnik Pamięci zwiększony o jeden
3B4A	#4A zostanie wpisane
22X	#22 zostanie wpisane i wykona się komenda X

P

wpisuje do obszaru pamięci określoną wartość.

Adresy początku i końca obszaru pamięci podajemy w odpowie-

dzi na "First:" (pierwszy bajt) i "Last:" (ostatni bajt). Następnie po "With:", podajemy wartość, którą chcemy wpisać, np. :

```
P
First: A000 ENTER     adres początku obszaru pamięci
Last: A200 ENTER      adres końca obszaru pamięci
With: FF ENTER        wartość
```

W komórki o adresach od #A000 do #A200 włącznie zostanie wpisana liczba #FF.

<u>Y</u>

Wpisuje znaki ASCII do pamięci począwszy od adresu wskazywanego przez Wskaźnik Pamięci.

Y podaje nową linię gdzie możemy wpisywać znaki ASCII bezpośrednio z klawiatury. Liczby odpowiadające tym znakom zostaną wpisane do pamięci począwszy od adresu we Wskaźniku Pamięci. Wpisując znaki możemy używać DELETE (CAPS SHIFT Ø) w celu skasowania znaku a cały ciąg należy zakończyć wciskając ⇐(CAPS SHIFT 5). Po wykonaniu komendy Y, Wskaźnik Pamięci zostanie ustawiony po ostatnim znaku wpisanego ciągu.

<u>I</u>

inteligentne kopiowanie

Blok pamięci zostanie skopiowany z jednego miejsca na drugie. Kopiowanie jest inteligentne w tym sensie, że blok może być kopiowany na obszar pamięci zachodzący na ten blok.
W odpowiedzi na "First:" podajemy adres początku bloku, który chcemy skopiować, a po "Last:", adres końca bloku tzn. adres ostatniego bajtu tego bloku. Następnie wpisujemy adres początku obszaru pamięci, na który chcemy podany blok sko-

piować (w odpowiedzi na "To:"). Adresy należy podawać w postaci liczb szesnastkowych. Jeśli adres początku bloku jest większy niż adres końca - komenda nie zostanie wykonana. Przykład:

```
I
First: C335
Last: C339
To: C337
```

	00	C334		00	C334
First →	E1	C335		E1	C335
	E2	C336		E2	C336
To →	E3	C337	To →	E1	C337
	E4	C338		E2	C338
Last →	E5	C339		E3	C339
	00	C33A		E4	C33A
	00	C33B		E5	C33B
	00	C33C		00	C33C

2.7 Modyfikacja zawartości rejestrów

<u>.</u>

Jeśli wpisaną liczbę szesnastkową zakończymy "." (SYMBOL SHIFT M), to zostanie ona wpisana do rejestru oznaczonego Wskaźnikiem Rejestru ">". Po wejściu do programu MONS3M21, Wskaźnik Rejestru wskazuje licznik rozkazów (PC).

Wskaźnik Rejestru możemy przesuwać za pomocą "." (bez liczby). Będzie on wskazywał kolejne rejestry oprócz SP i IR, których zawartości nie możemy zmienić.

Przykłady:

założmy, że Wskaźnik Rejestru jest ustawiony na PC

. przesuń Wskaźnik Rejestru na IY

.	przesuń Wskaźnik Rejestru na IX
Ø.	wpisz Ø do rejestru IX
.	przesuń Wskaźnik Rejestru na HL
1ABF.	wpisz #1ABF do HL
.	przesuń Wskaźnik Rejestru na DE
.	przesuń Wskaźnik Rejestru na BC
7FF.	wpisz #07FF do BC
.	przesuń Wskaźnik Rejestru na AF
FF00	wpisz #FF do rejestru A i wyzeruj wszystkie znaczniki stanu
.	przesuń Wskaźnik Rejestru na PC
9200.	ustaw licznik rozkazów na adres #9200

Q

zmienia zestaw wyświetlanych rejestrów

Po wejściu do programu MONS3M21, na tablicy wyświetlany jest zestaw rejestrów podstawowych (tzn. HL, DE, BC, AF). Po naciśnięciu Q, wyświetlony zostanie zestaw rejestrów dodatkowych (tzn. HL', DE', BC', AF'), wyróżnionych apostrofem "'". Ponowne użycie komendy Q przywróci wyświetlanie podstawowego zespołu rejestrów roboczych. Zawartość rejestrów dodatkowych możemy również modyfikować za pomocą ".".

2.8 Zmiana zapisu liczb

SYMBOL SHIFT 3 #

zmienia wyświetlane adresy z zapisu szesnastkowego na dziesiętny lub odwrotnie. Dotyczy to również zapisu adresów podawanych podczas deasemblacji kodu oraz wyświetlania zawar-

tości pamięci.

Liczby pokazujące zawartość komórek pamięci są zawsze wyświetlane w zapisie szesnastkowym.

H

Zamienia liczbę dziesiętną na szesnastkową.

Po ":" wprowadzamy liczbę dziesiętną zakończoną dowolnym znakiem oprócz cyfry (Ø-9). W tej samej linii wyświetlony zostanie znak "=" oraz liczba szesnastkowa. Po naciśnięciu dowolnego klawisza możemy wprowadzać następne komendy.

Przykład:

H:43765 = AAF5

Czasami będziemy potrzebowali zamienić liczbę szesnastkową na dziesiętną. Najprostszym sposobem dokonania takiej zamiany jest wpisanie liczby szesnastkowej do któregoś z rejestrów a następnie użycie komendy SS 3 aby zamienić ją na liczbę dziesiętną.

2.9 Powrót do systemu operacyjnego ZX Spectrum

CAPS SHIFT 1 EDIT

Komenda ta może być wykonana w dowolnym momencie pracy programu MONS3M21

Aby ponownie wejść do programu MONS3M21 należy wpisać:

RANDOMIZE USR (adres początku programu)

a w przypadku programu MONS3:

RANDOMIZE USR (adres początku programu + 29)

ROZDZIAŁ 3. PRZYKŁAD UŻYCIA PROGRAMU MONS3M21

W przykładzie wykorzystamy program podany w załączniku instrukcji do programu GENS3M21. Kod wynikowy umieszczony został od adresu #D000.

Oto pełny tekst programu:

```
D000  ED4B33D0          LD   BC,(LD033)   ; część I
D004  ED5B35D0          LD   DE,(LD035)
D008  CD37D0            CALL LD037
D00B  7A                LD   A,D
D00C  CD1CD0            CALL LD01C
D00F  7B                LD   A,E
D010  CD1CD0            CALL LD01C
D013  7C                LD   A,L
D014  CD1CD0            CALL LD01C
D017  7D                LD   A,H
D018  CD1CD0            CALL LD01C
D01B  C9                RET

D01C  F5        LD01C   PUSH AF             część II
D01D  0F                RRCA
D01E  0F                RRCA
D01F  0F                RRCA
D020  0F                RRCA
D021  CD25D0            CALL LD025
D024  F1                POP  AF
D025  E615      LD025   AND  #15
D027  C690              ADD  A,#90
D029  27                DAA
D02A  CE40              ADC  A,#40
D02C  27                DAA
D02D  FD213A5C          LD   IY,#5C3A
D031  D7                RST  #10
D032  C9                RET

D033  409C      LD033   DEFB "@",#9C      ; mnożna
D035  FA00      LD035   DEFB #FA,#0       ; mnożnik
```

```
D037 210000  LD037  LD   HL,#0000    część III
D03A 3E10           LD   A,#10
D03C 29      LD03C  ADD  HL,HL
D03D EB             EX   DE,HL
D03E ED6A           ADC  HL,HL
D040 EB             EX   DE,HL
D041 3004           JR   NC,LD047
D043 09             ADD  HL,BC
D044 3001           JR   NC,LD047
D046 13             INC  DE
D047 3D      LD047  DEC  A
D048 20F2           JR   NZ,LD03C
D04A C9             RET
```

Chcemy zobaczyć jak działa procedura znajdująca się w trzeciej części kodu. Procedura ta mnoży dwie liczby 16-bitowe bez znaku. Liczby te znajdują się w parach rejestrów BC i DE. 32--bitowy wynik mnożenia umieszczany jest w rejestrach DE (bardziej znaczące 16 bitów) i HL (mniej znaczące 16 bitów).
Fragment kodu w części pierwszej przygotowuje parametry dla procedury mnożącej, wywołuje ją a następnie wywołuje procedurę z części drugiej wyprowadzającą zawartość rejestru A (w postaci liczby szetnastkowej) na ekran. Procedura ta wywoływana jest cztery razy dla każdego z rejestrów, w których umieszczony jest wynik mnożenia (HLDE).
Działanie programu możemy sprawdzić za pomocą poniższych komend:

M: D000 ENTER	ustawiamy Wskaźnik Pamięci na początek programu
D000.	wpisujemy ten adres do licznika rozkazów PC
SS Z	wykonujemy jeden rozkaz
SS Z	jw - do rejestrów BC i DE wpisane zostały mnożna i mnożnik
SS Z	wywołujemy procedurę mnożącą
SS Z	wpisujemy Ø do HL

SS Z	i #10 do A (licznik bitów)
SS Z (4 razy)	DEHL zostało przesunięte w lewo
SS Z	skok do #D047
SS Z	zmniejszamy licznik bitów
SS Z	skok z powrotem itd.
	po zorientowaniu się jak działa ten fragment:
,	ustawiamy Wskaźnik Pamięci na adres powrotu z procedury mnożącej
W	wpisujemy adres powrotu w tym miejscu
SS K	kontynuujemy wykonywanie procedury
	po sygnale dźwiękowym naciskamy dowolny klawisz; w rejestrach DEHL znajduje się wynik mnożenia
SS Z	wpisujemy pierwsze 8 bitów wyniku do akumulatora
SS Z	wywołujemy procedurę wyprowadzającą zawartość akumulatora na ekran
,	ustawiamy Wskaźnik Pamięci na adres na stosie
W	i wpisujemy adres powrotu
SS K	wykonujemy całą drugą część
	na ekranie po znaku "+" pojawi się "00" - zawartość rejestru A
SS Z	wpisujemy następne 8 bitów wyniku do rejestru A
SS T	wpisujemy adres powrotu po bieżącym rozkazie i wykonujemy całą procedurę itd.

Możemy również sprawdzić działanie samej procedury mnożącej dla dowolnie wybranych liczb, np. 50000 i 60000:

M: D037	ustawiamy Wskaźnik Pamięci na początek procedury mnożącej
D037.	wpisujemy ten adres do PC
.	przesuwamy Wskaźnik Rejestru
.	- " -

.	- " -
.	Wskaźnik Rejestru wskazuje na DE
H:50000 ENTER	zamieniamy pierwszą liczbę na postać szestnastkową
C350.	wpisujemy tę liczbę do DE
.	przesuwamy Wskaźnik Rejestru do BC
H:60000 ENTER	zamieniamy drugą liczbę
EA60.	i wpisujemy ją do BC
SS Z	oglądamy działanie procedury
	. . .
M:D04A ENTER	ustawiamy Wskaźnik Pamięci na koniec procedury
W	wpisujemy adres powrotu
SS K	kontynuujemy wykonywanie procedury do końca
	po naciśnięciu dowolnego klawisza możemy obejrzeć
	wynik: #B2D05E00 = 3000000000

ZAŁACZNIK   TABLICA MONS3

```
9092  2836        JR    Z,#90CA
>PC 9092   28 36 EB 09 D5 E5 5E
 SP 750F   2B 2D 65 33 58 27 ED
 IY 5C3A   FF ED 00 22 75 23 75
 IX 2152   04 0C 0D 20 FD 0E 3F
 HL 2152   CD 2C 0F FD CB 01 EE
 DE FF00   00 00 00 00 00 00 00
 BC 0053   E1 6E FD 75 00 EB 7B
 AF FFAC  S    V
 IR 3F00

9086  00   908E  56   9096  D5
9087  00   908F  23   9097  E5
9088  21   9090  7A   9098  5E
9089  F9   9091  B3   9099  23
908A  20  >9092  28<  909A  56
908B  09   9093  36   909B  EB
908C  5E   9094  EB   909C  09
908D  23   9095  09   909D  EB

>
```

W pierwszej linii, u góry ekranu, znajduje się adres pamięci, na który ustawiony jest Wskaźnik Pamięci (9092) oraz zawartość bajtów bieżącego rozkazu (2836) wraz z zapisem symbolicznym rozkazu (JR Z,#90CA).

W kolejnych dziewięciu liniach przedstawione są rejestry mikroprocesora Z80; nazwa rejestru (lub pary rejestrów) i jego zawartość, np.: HL 2152 oznacza, że w rejestrze H znajduje się #21, a w L - #52. Siedem kolumn liczb na prawo od rejestrów PC ÷ BC, to zawartości kolejnych komórek pamięci począwszy od adresu znajdującego się w rejestrze, np.: w rejestrze SP znajduje się wartość #9092, a #28 to zawartość komórki pamięci o tym adresie, #36 - komórki o adresie #9093, #EB - #9094 itd.

Zawartość rejestru znaczników stanu mikroprocesora (F) jest dekodowana - jeśli znacznik jest równy jeden, to jego symbol jest wyświetlany obok; W powyższym przykładzie S i V.

Symbole znaczników stanu są następujące:

S znak
Z zero
H przeniesienie pomocnicze
V parzystość lub nadmiar
N odejmowanie
C przeniesienie

Wskaźnik Rejestru, ">", wskazuje na rejestr PC.

Następna część Tablicy MONS3 przedstawia adresy 24 komórek pamięci (w przykładzie od #9086 do #909D) wraz z ich zawartością. Adres we Wskaźniku Pamięci jest oznaczony przez "> <".

Znak ">", u dołu ekranu, wskazuje na miejsce gdzie wprowadzane są komendy. Tablica MONS3 jest uaktualniana po wykonaniu każdej komendy.

INDEKS KOMEND PROGRAMU MONS3M21

1. Ustawianie Wskaźnika Pamięci (WP)

M aa	WP:=aa
ENTER	WP:=WP+1
CS 8 ⇒	WP:=WP+8
CS 7 ⇑	WP:=WP-1
CS 5 ⇐	WP:=WP-8
X	WP_x:=WP , WP_l:=M(WP) , WP_h:=M(WP+1)
V	WP:=WP_x
O	WP_o:=WP , WP:=WP+M(WP)+1
U	WP:=WP_o
,	WP_l:=M(SP) , WP_h:=M(SP+1)

2. Deasemblacja kodu

SS 4 $	deasemblacja od adresu w WP
T aa	deasemblacja od adresu aa - również: na drukarkę oraz generowanie tekstu dla programu GENS3M21

3. Wykonywanie programów w kodzie maszynowym

SS Z :	wykonywanie kodu po jednym rozkazie od adresu w WP
W	wpisywanie adresu powrotu w miejsce wskazywane przez WP
SS K +	wykonywanie kodu od adresu w liczniku rozkazów (PC)
SS T >	wykonywanie kodu od adresu w WP po uprzednim wpisaniu adresu powrotu po bieżącym rozkazie
J aa	wykonywanie kodu od adresu aa

4. Wyświetlanie zawartości pamięci

L	wyświetlanie zawartości pamięci od adresu w WP
SS P "	Jw ale wydruk skierowany na drukarkę

5. Przeszukiwanie pamięci

G $n_1, n_2, \ldots$	przeglądanie pamięci w poszukiwaniu ciągu liczb $n_1, n_2, \ldots$
N	dalsze przeszukiwanie

6. Modyfikacja zawartości pamięci

n	M(WP):=n
P	wpisywanie podanej liczby do obszaru pamięci
Y	wpisywanie znaków ASCII do pamięci od adresu w WP
I	kopiowanie bloku pamięci

7. Modyfikacja zawartości rejestrów

.	przesuwanie Wskaźnika Rejestru (WR)
nn.	R(WR):=nn
Q	zmiana zestawu wyświetlanych rejestrów

8. Zmiana zapisu liczb

SS 3 #	zmiana wyświetlanych adresów z systemu szestnastkowego na dziesiętny lub odwrotnie
H d	zamiana podanej liczby dziesiętnej d na liczbę szestnastkową

9. Powrót do systemu operacyjnego ZX Spectrum

CS 1 EDIT

oznaczenia: WP - Wskaźnik Pamięci
M(WP) - zawartość komórki pamięci o adresie WP
M(SP) - zawartość komórki pamięci o adresie we wskaźniku stosu SP
R(WR) - zawartość rejestru wskazywanego przez Wskaźnik Rejestru WR
aa - adres 16-bitowy
n - liczba 8-bitowa
nn - liczba 16-bitowa
d - liczba dziesiętna

SPIS TREŚCI

Stołeczny
Ośrodek
Elektronicznej
Techniki
Obliczeniowej
SOETO

00-682 WARSZAWA, ul. Hoża 50
telefon: 218326
telex: 814786

Wykonuje:

Usługi informatyczne
na bazie sprzętowej
komputerów serii ODRA i RIAD

Usługi w zakresie
informatyki mikrokomputerowej

SOETO — STUDIO
MIKROKOMPUTEROWE "BIT"

00-060 WARSZAWA, ul. Królewska 27
telefon: 277281 w. 526

— realizuje różne formy szkolenia
— wydaje materiały szkoleniowe
— prowadzi „SALON GIER"
— wykonuje usługi obliczeniowe
realizując hasło:

MIKROKOMPUTERY:
— UCZĄ
— BAWIĄ
— PRACUJĄ

Cena 360 zł

OE
TO
SM
„BIT"



DRUK: Z.P. WOŁOMIN UL. Wł. SIKORSKIEGO 70
ZAM. 883/86 NAKŁAD 1000 EGZ. P-23

